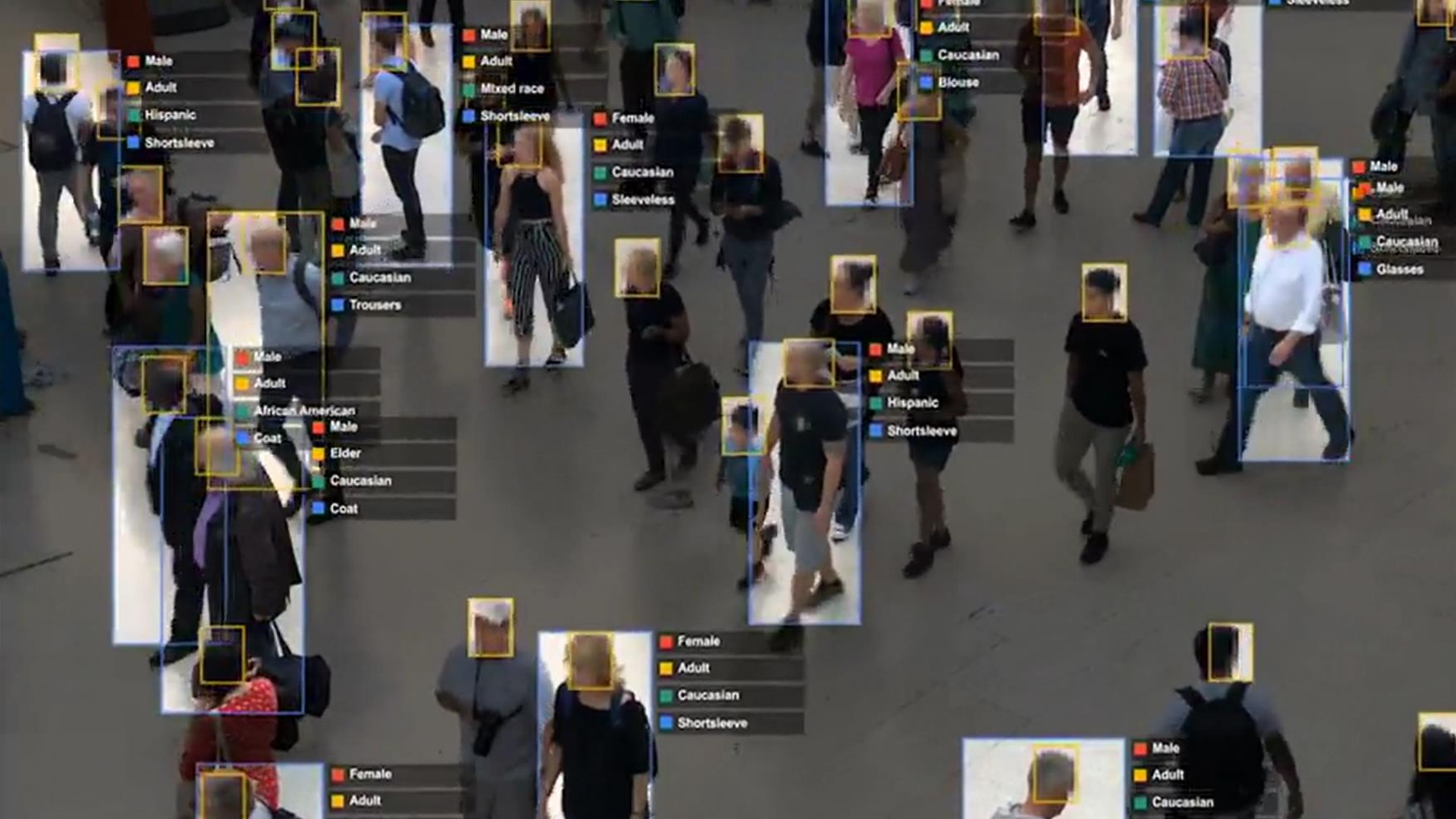
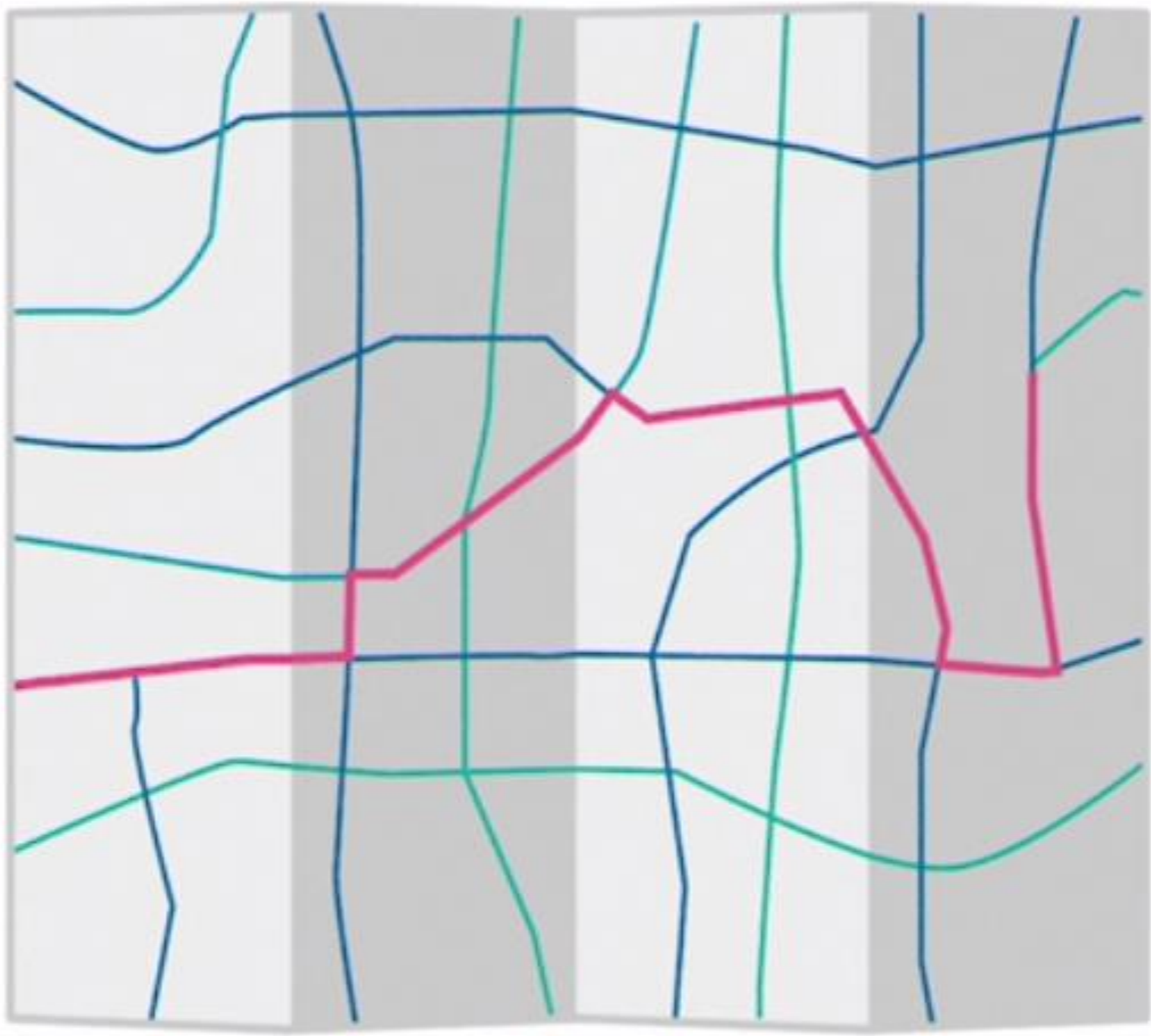
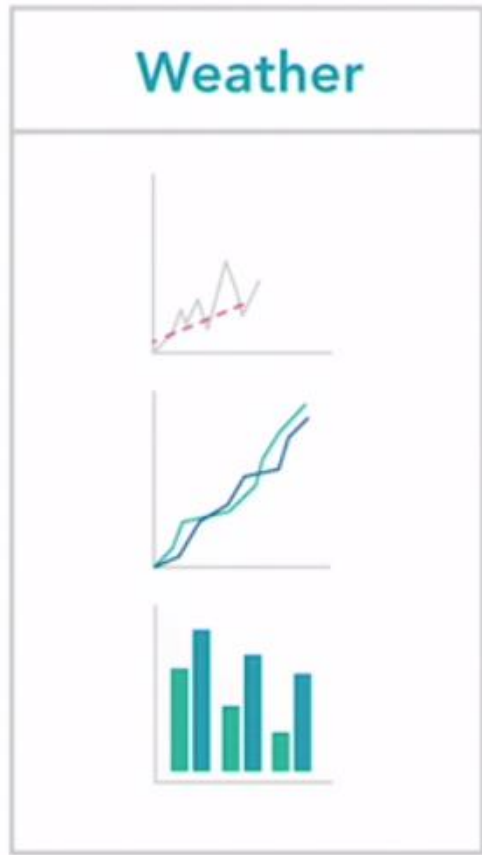

Male
Adult
Hispanic
Shortsleeve
Male
Adult
Mixed race
Shortsleeve
Female
Adult
Caucasian
Sleeveless
Male
Adult
Caucasian
Blouse
Male
Male
Adult
Caucasian
Glasses
Male
Adult
Caucasian
Trousers
Male
Adult
African American
Coat
Male
Elder
Caucasian
Coat
Male
Adult
Hispanic
Shortsleeve
Female
Adult
Caucasian
Shortsleeve
Female
Adult
Male
Adult
Caucasian

Ready



Done

Test Data

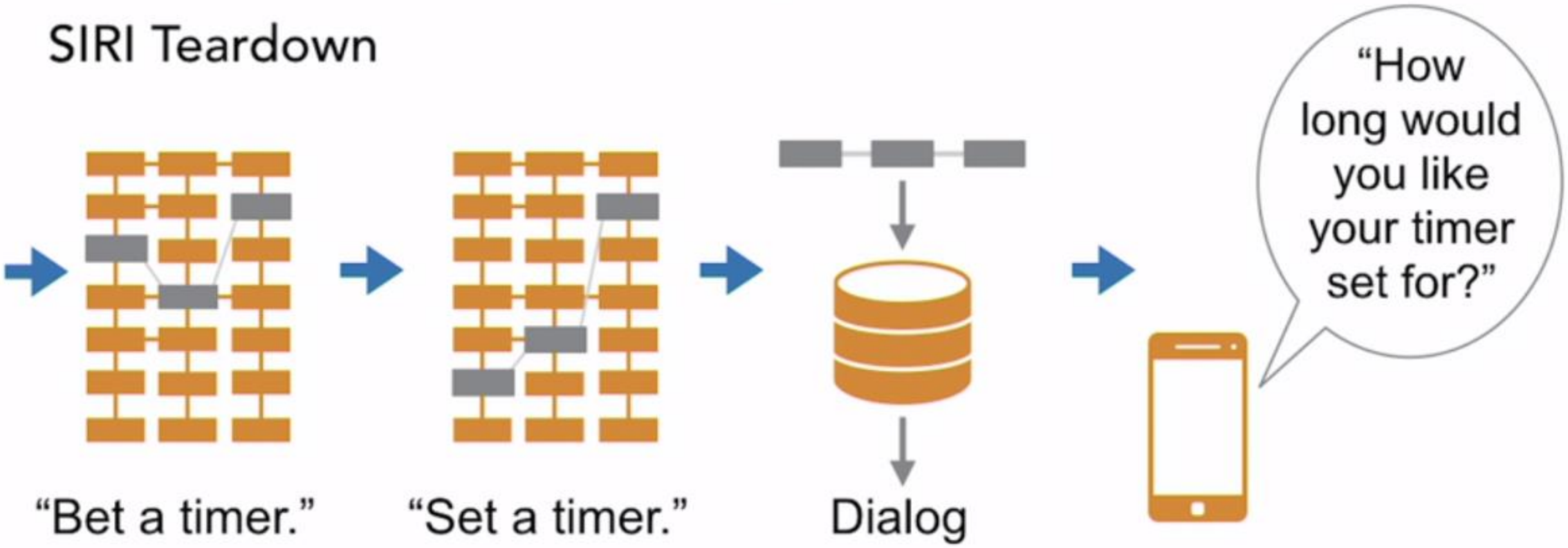


Machine Learning Fundamentals

Machine Learning is all around you!

Personal assistants – Siri, Google Home, Amazon Echo

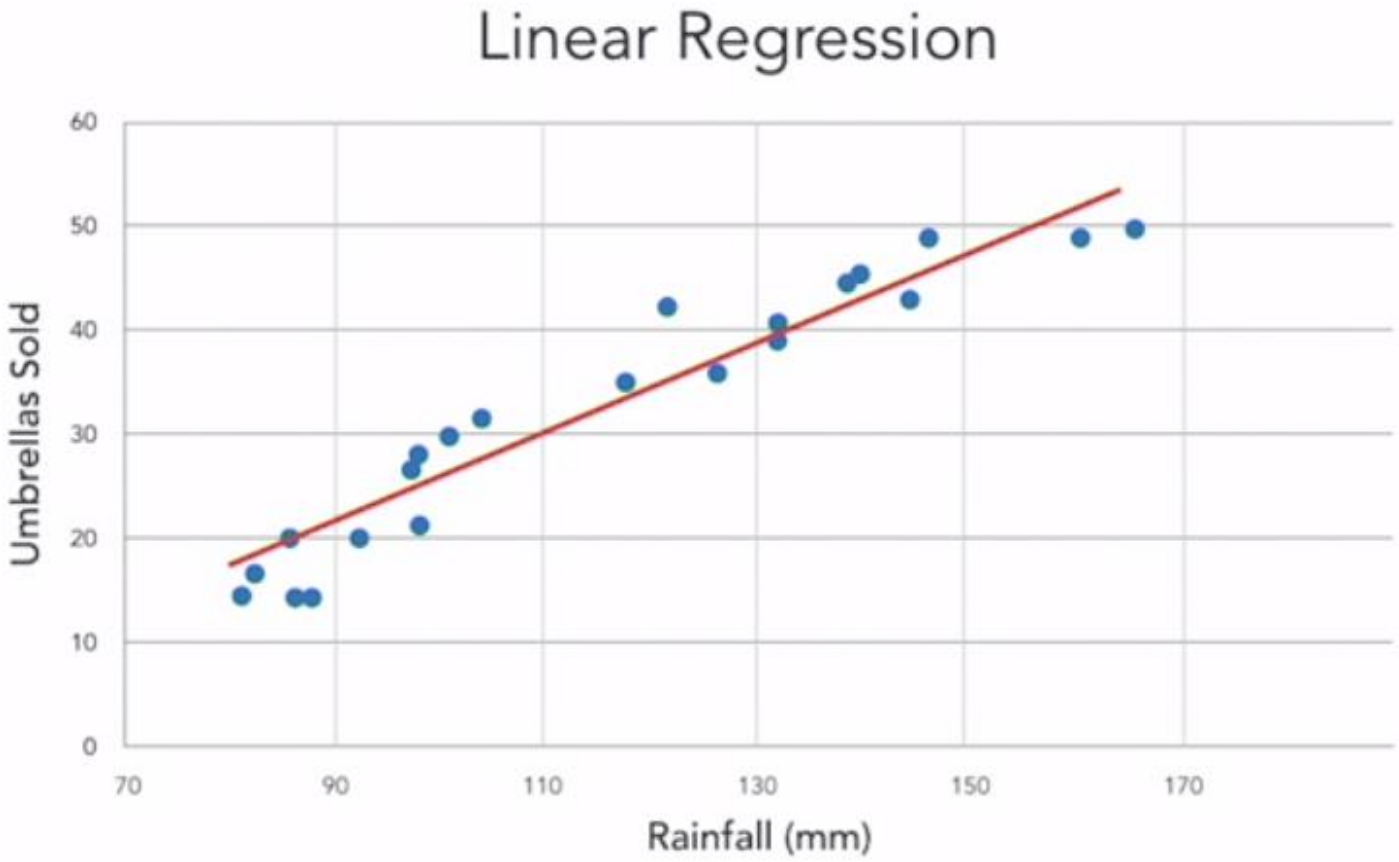
SIRI Teardown



What is Machine Learning ?

- Pattern matching

$$y = mx + b$$

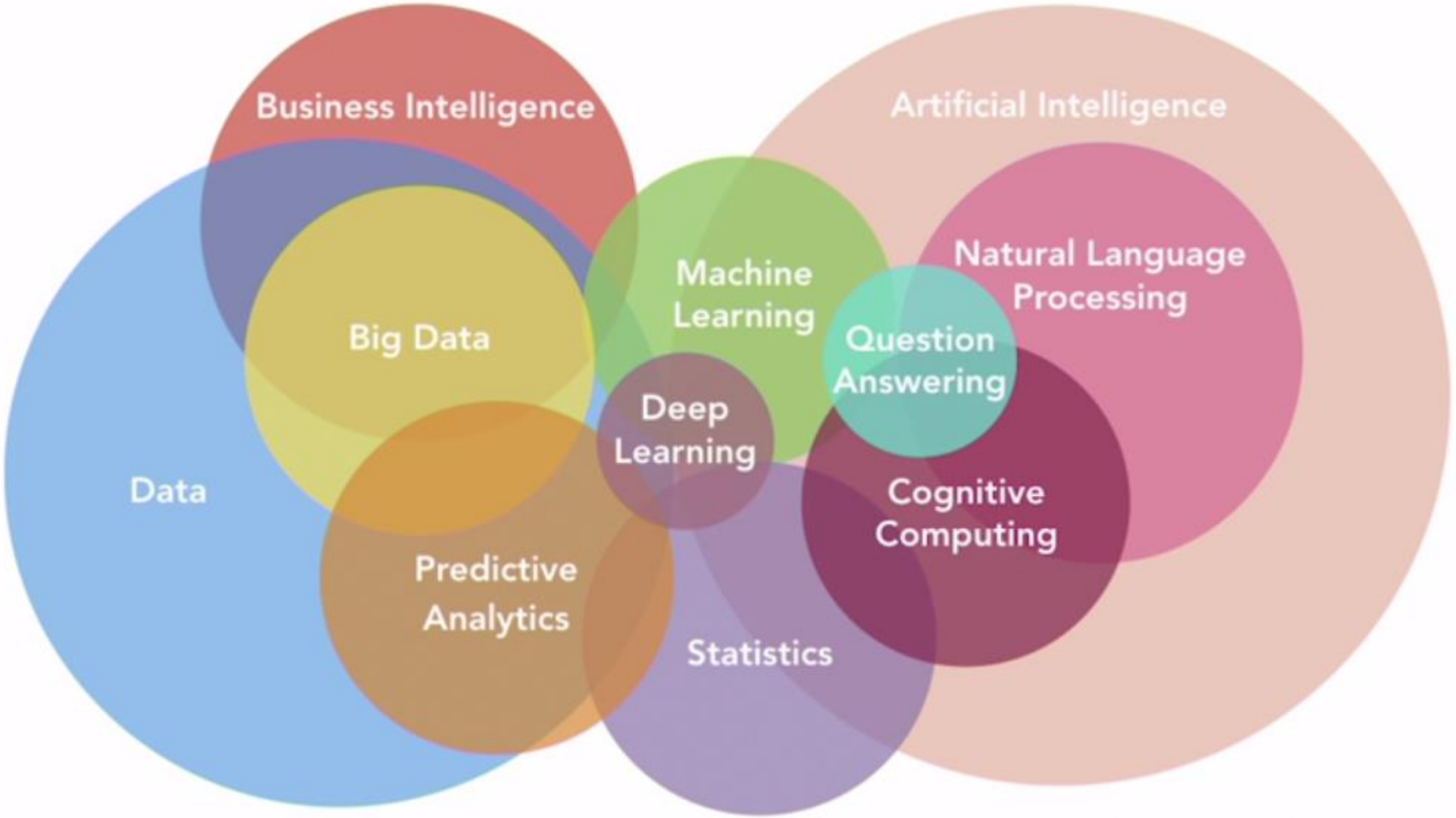



EXIT


27cn4X0007U1901
9c0U43X092209728
2X13722n5478373
711y70591F8t910
Z99440ha1921291


elseif dir == "left" then
inv = true
changeAcc()
elseif dir == "right" then
inv = false
changeAcc()
end
elseif selectedObj==randomRect then
randomMonster()
elseif selectedObj==backRect then
if dir == "left" then
inv = true
changeBackground()
if dir == "right" then
inv = false
changeBackground()
end
end
display.getCurrentStage():setFocus(nil)
end
----BLINKING-----
hideBlink = function()
eyes.isVisible=true
iris.isVisible=true
blinking.hide()
blinkingPart.hide()
end


1.83
2.6
3.07
1.63
117.52
1.01
0.91
1.94
29.18
15.36
2.53
1.42
156.11
3.64
1.59
28.58
3.43
3.64
3.07
97.71
07
1.5
72.88
1.33
1.43
110.33
2.97
2.27
45.55
1.44
-6.02


Business Intelligence
Artificial Intelligence
Big Data
Machine Learning
Natural Language Processing
Question Answering
Deep Learning
Data
Cognitive Computing
Predictive Analytics
Statistics

This lecture will cover some of the foundations of machine learning like:

- exploratory data analysis
- cleaning your data
- fitting robust models
- tuning hyperparameters
- and finally, evaluating a model to ensure that it generalizes to unseen examples.

What You Should Know?

- Basic Python Knowledge
- Basic Experience with Numpy, Pandas and Scikit-learn

Good to Have, But not Required!

- Basic Statistics
- Entry Level Data Analysis Experience
- Familiarity with common Machine Learning Terms

Tools we would be using

- Python 3.4
- Jupyter Notebook

Jupyter Notebooks is a fantastic tool that allows you to combine:

- live code,
- explanatory text,
- visualizations,
- and equations all in one concise package.

"Field of study that gives computers the ability to learn without being explicitly programmed."

—Arthur Samuel, IBM, 1959

"Machine learning algorithms can figure out how to perform important tasks by generalizing from examples."

—Pedro Domingos, University of Washington

Fitting a function to examples and using that function to generalize and make predictions about new examples

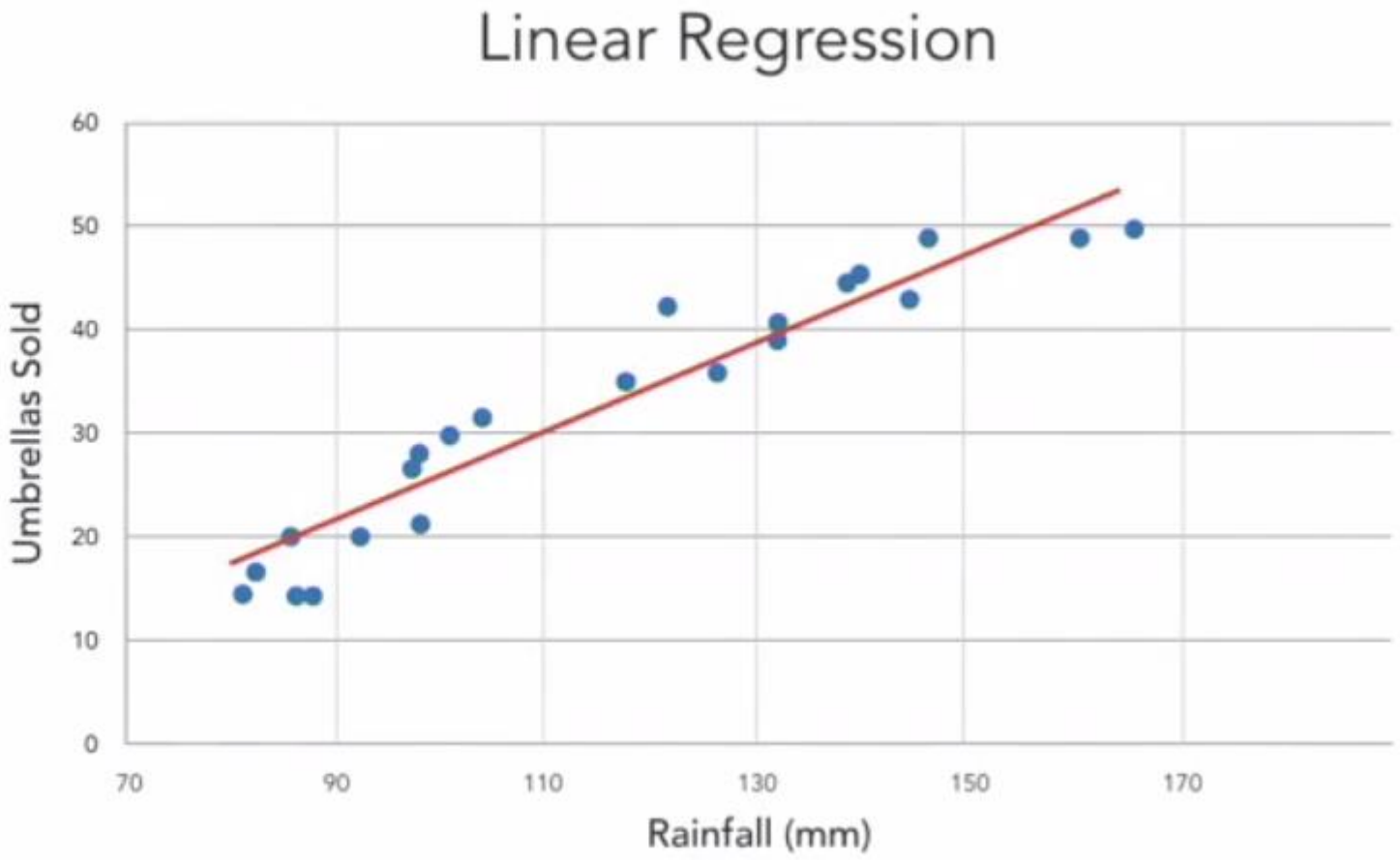
What is Machine Learning?

- Pattern Matching

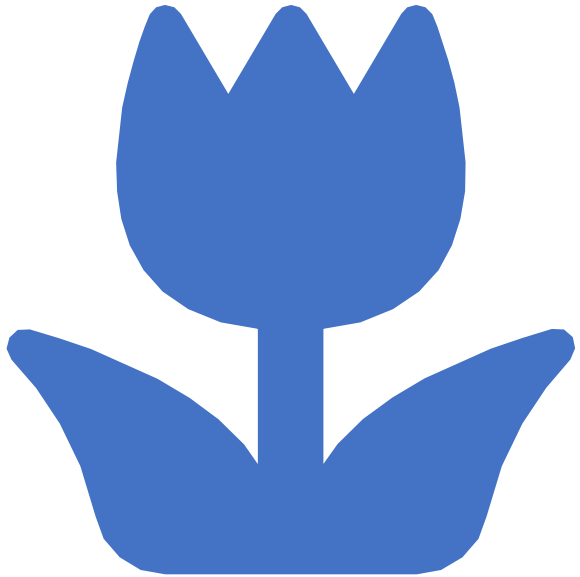
What is Machine Learning?

- Pattern matching

$$y = mx + b$$



150 Flowers

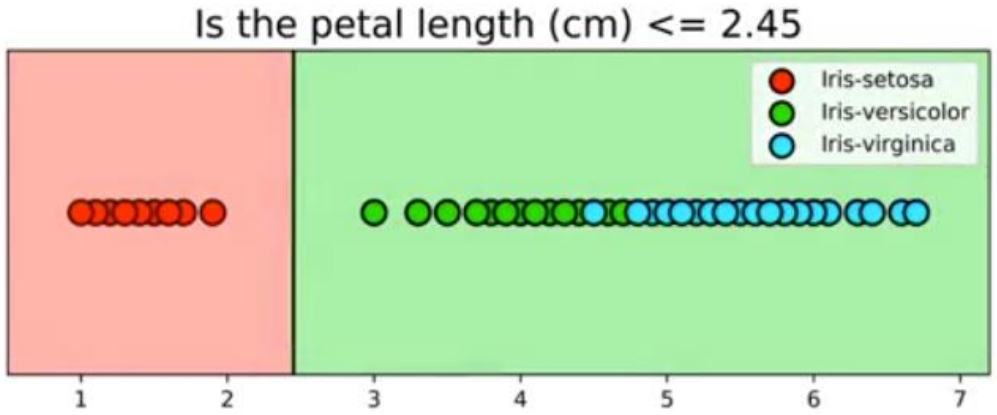


- Determine flower species by features such as petal length

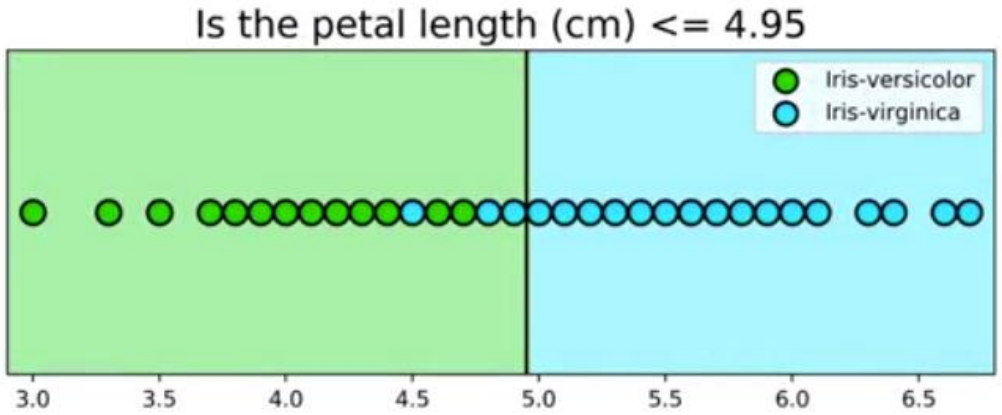
SETOSA

VERSICOLOR

VIRGINICA



Separate Setosa from other Species



Separate Versicolor from other Virginica

You could keep writing rules until you completely separate out the flower species

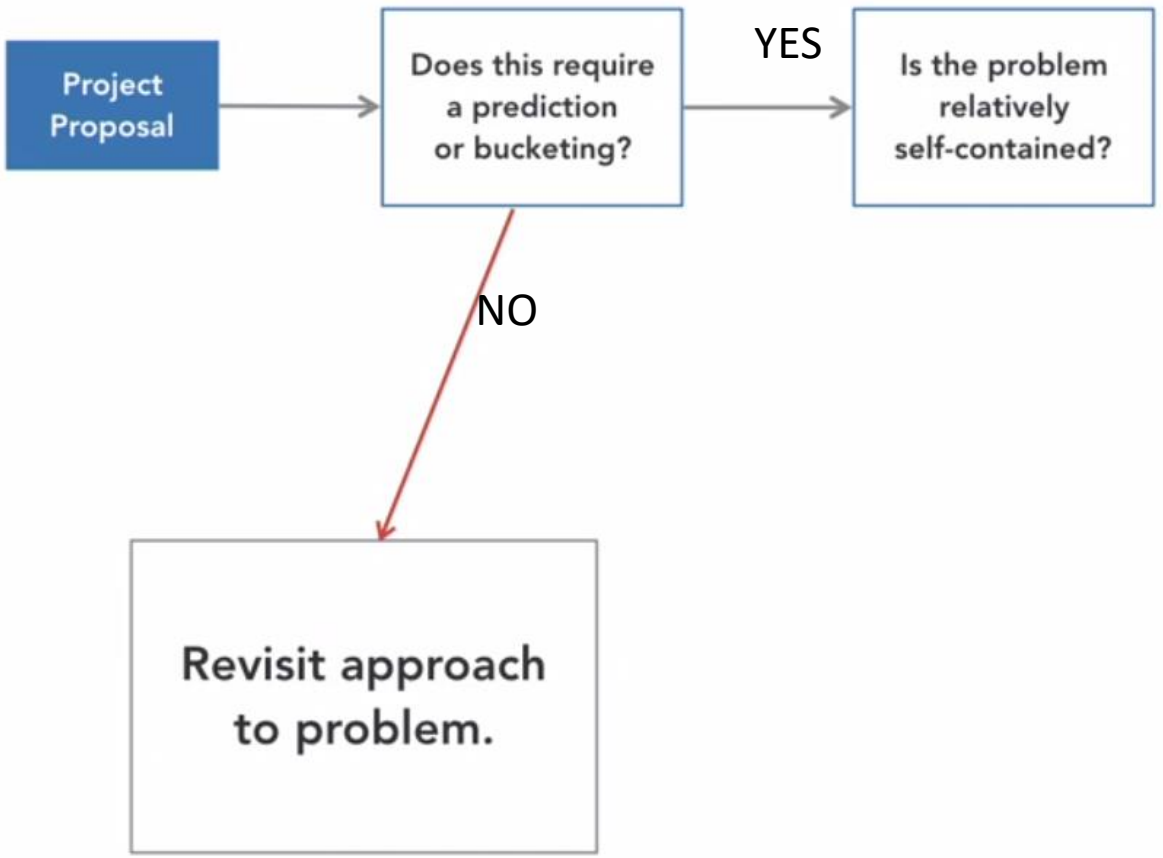
Assess the Problem You're Trying to Solve

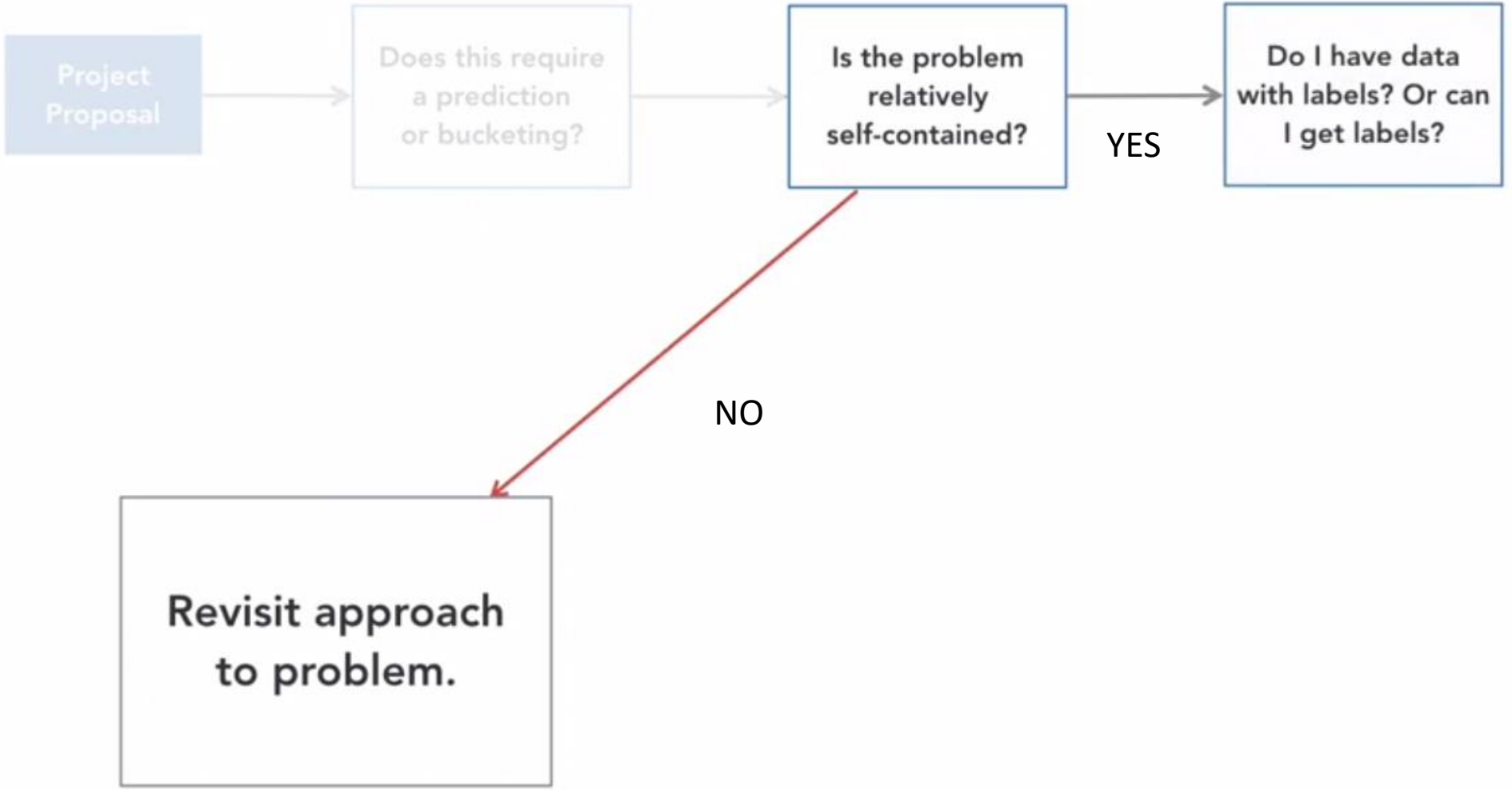
Is this a type of problem that can be solved using machine learning?

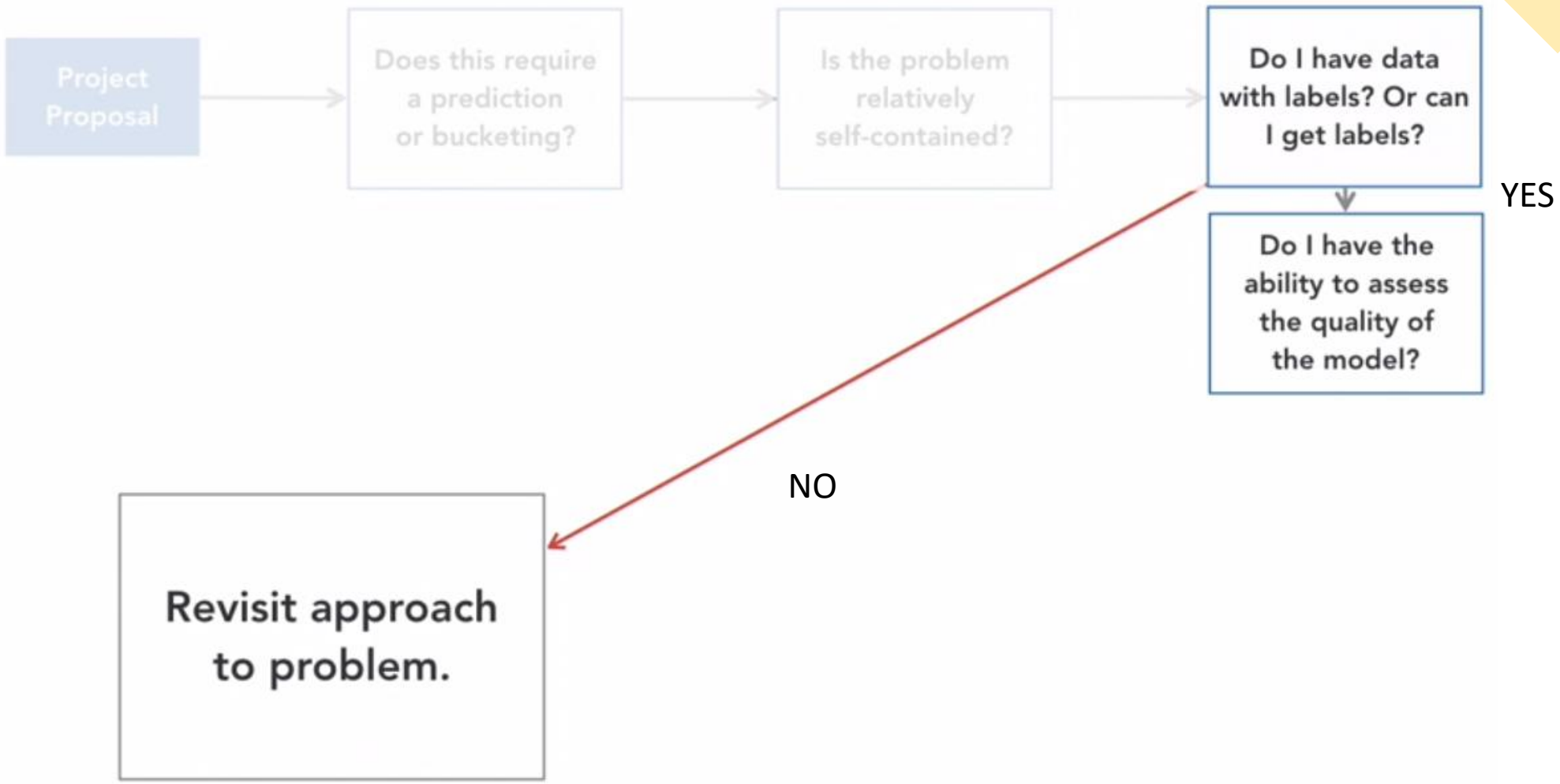
- Does it require a prediction or bucketing something into categories?
- Is the problem relatively self-contained?

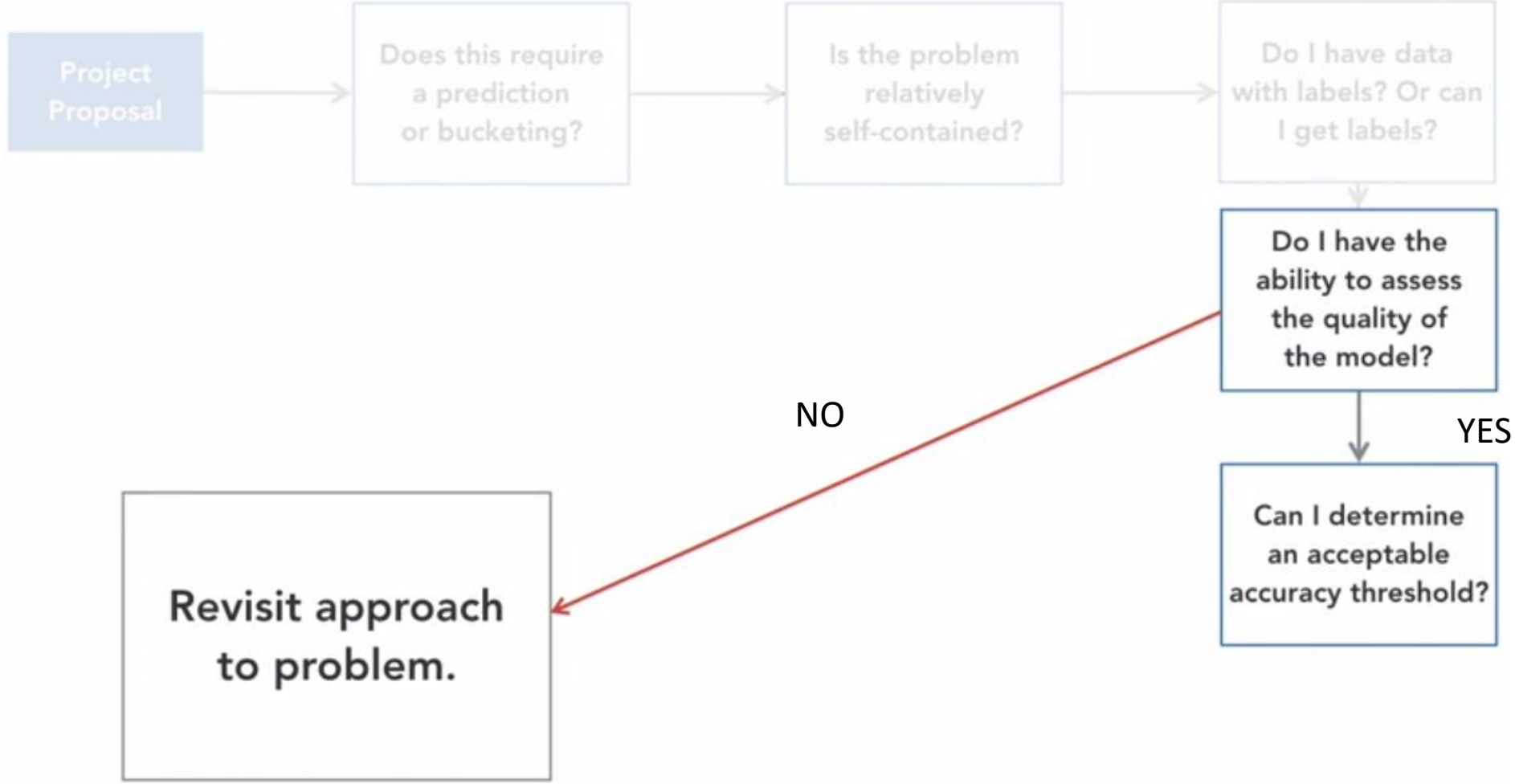
Is my particular problem one that can be solved with machine learning?

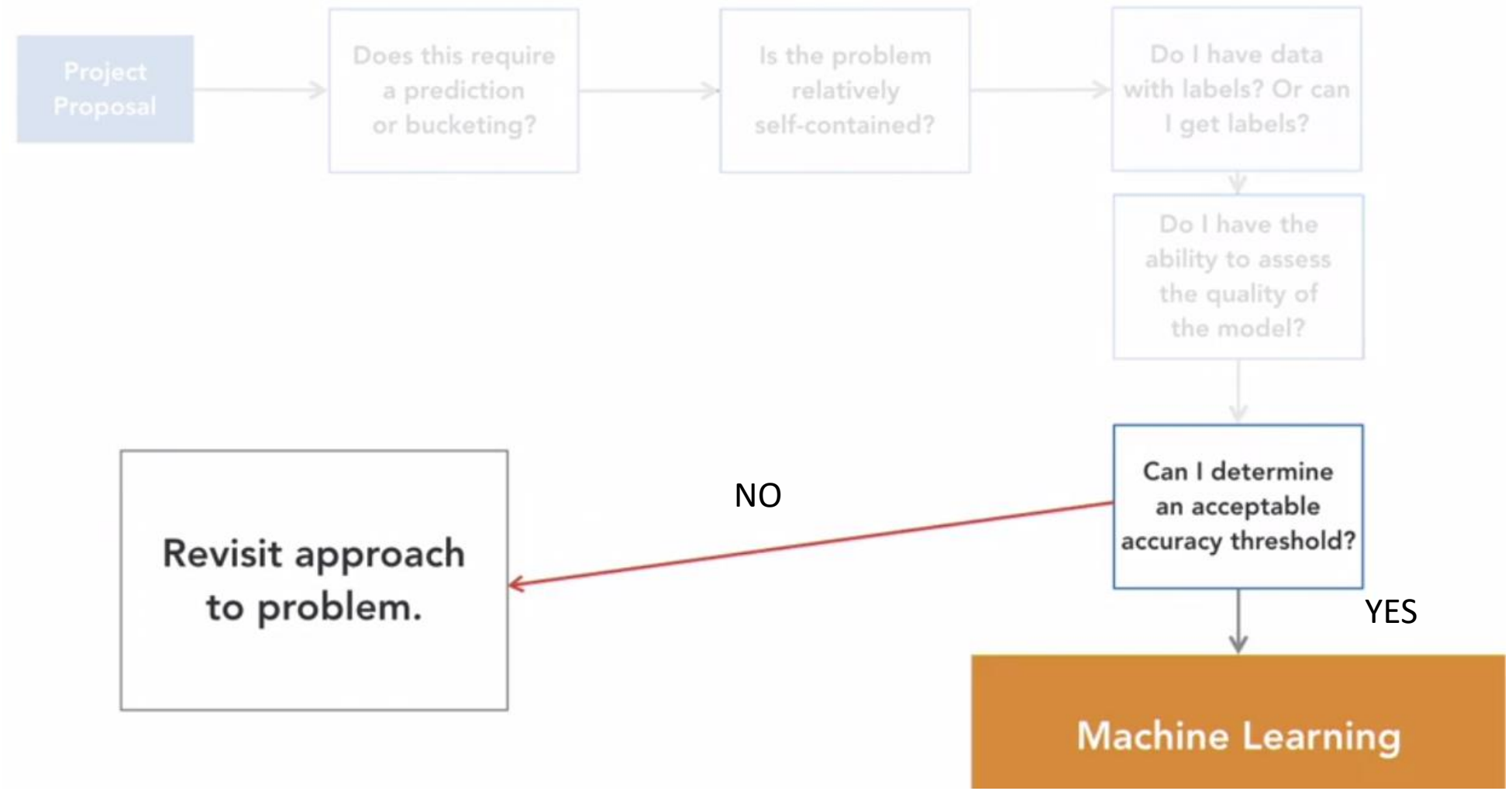
- Do I have data with labels?
- Do I have the ability to assess the quality of the model?
- Can I determine an acceptable accuracy threshold?


Project Proposal
Does this require a prediction or bucketing?
YES
Is the problem relatively self-contained?
NO
Revisit approach to problem.


Project Proposal
Does this require a prediction or bucketing?
Is the problem relatively self-contained?
YES
Do I have data with labels? Or can I get labels?
NO
Revisit approach to problem.


Project Proposal
Does this require a prediction or bucketing?
Is the problem relatively self-contained?
Do I have data with labels? Or can I get labels?
YES
Do I have the ability to assess the quality of the model?
NO
Revisit approach to problem.


Project Proposal
Does this require a prediction or bucketing?
Is the problem relatively self-contained?
Do I have data with labels? Or can I get labels?
Do I have the ability to assess the quality of the model?
NO
YES
Can I determine an acceptable accuracy threshold?
Revisit approach to problem.


Project Proposal
Does this require a prediction or bucketing?
Is the problem relatively self-contained?
Do I have data with labels? Or can I get labels?
Do I have the ability to assess the quality of the model?
Can I determine an acceptable accuracy threshold?
NO
Revisit approach to problem.
YES
Machine Learning

Examples of machine learning in everyday life

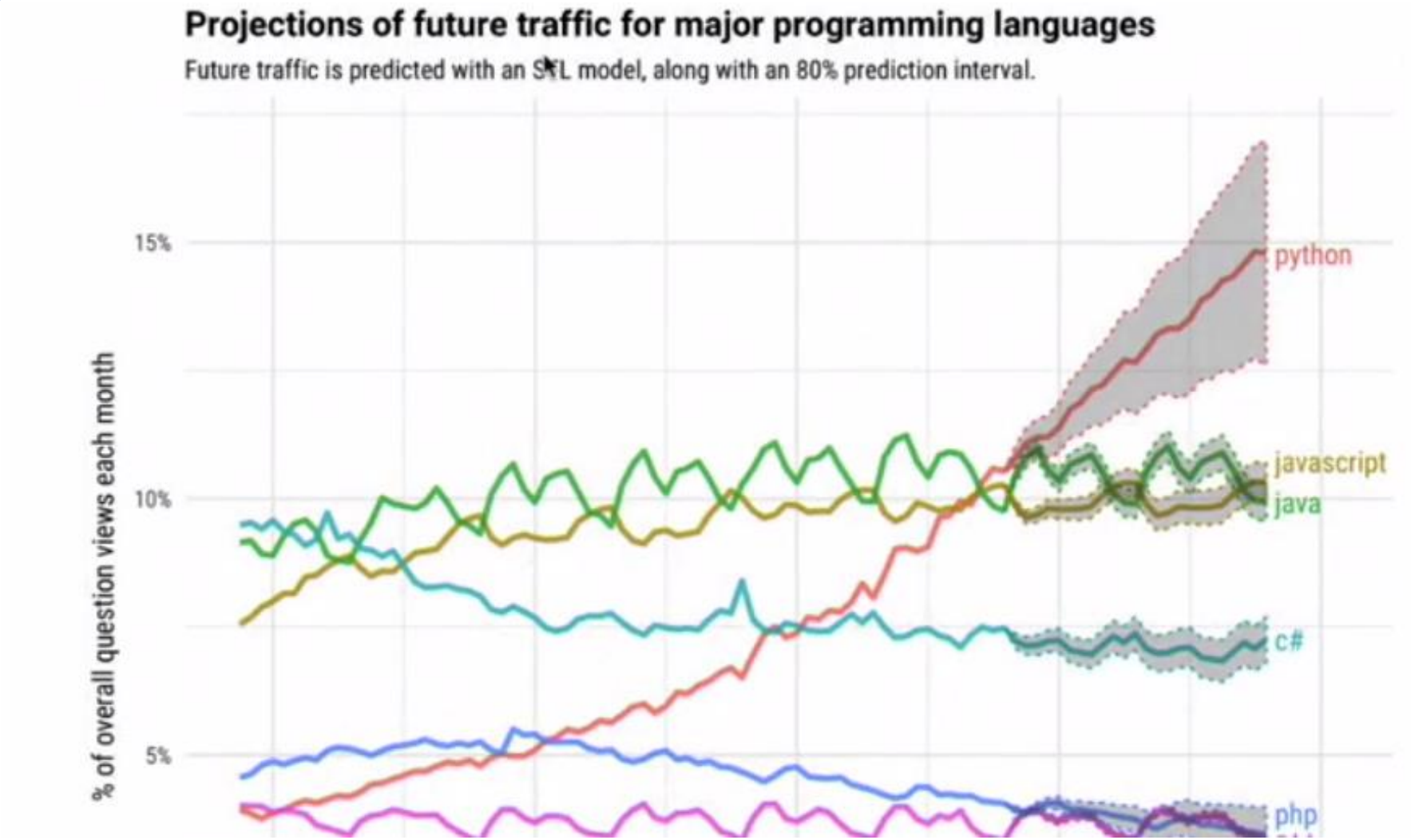
- spam detection
- speech recognition
- recommendations
- fraud detection.

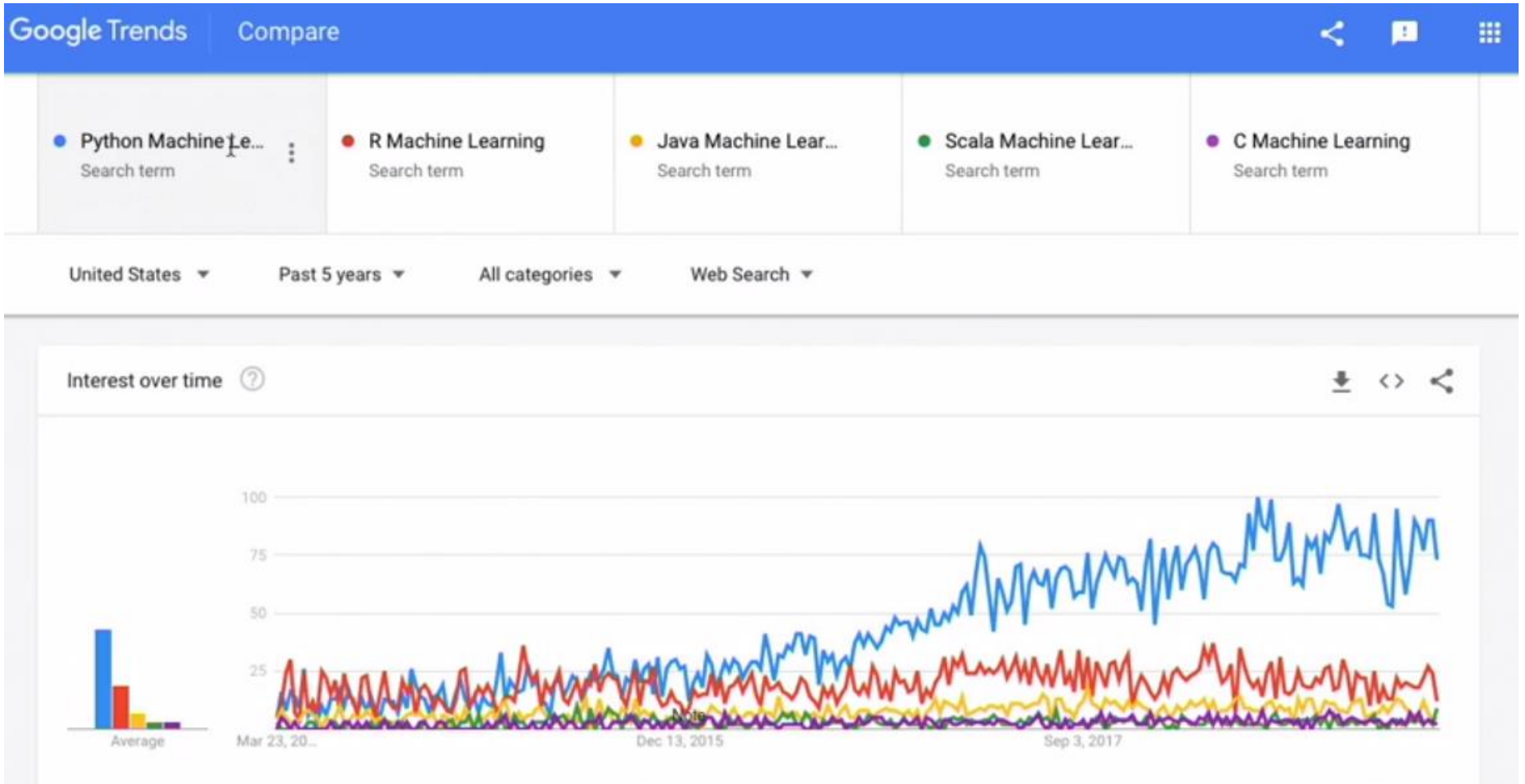
Why we will be using Python for Machine Learning?

Popular right now, with a large user base

More machine learning libraries/packages than for any other language

Easy to learn & use


Projections of future traffic for major programming languages
Future traffic is predicted with an STL model, along with an 80% prediction interval.
% of overall question views each month
15%
10%
5%
python
javascript
java
c#
php


Google Trends
Compare
Python Machine Le...
Search term
R Machine Learning
Search term
Java Machine Lear...
Search term
Scala Machine Lear...
Search term
C Machine Learning
Search term
United States
Past 5 years
All categories
Web Search
Interest over time
100
75
50
25
Average
Mar 23, 20...
Dec 13, 2015
Sep 3, 2017

Python Is a Swiss Army Knife

- Machine learning – scikit-learn, LightGBM, SciPy
- Deep learning – TensorFlow, PyTorch, keras
- Text – NLTK, genism

Python Is a Swiss Army Knife

- Images –
  OpenCV, scikit-learn
- Visualization –
  Matplotlib, Seaborn
- Data analysis –
  pandas, NumPy

In [2]:
```
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)

from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
```

In [3]:
```
iris = load_iris()
```

In [4]:
```
model = LogisticRegression().fit(iris['data'], iris['target'])
```

In [6]:
```
model.predict(iris['data'])
```

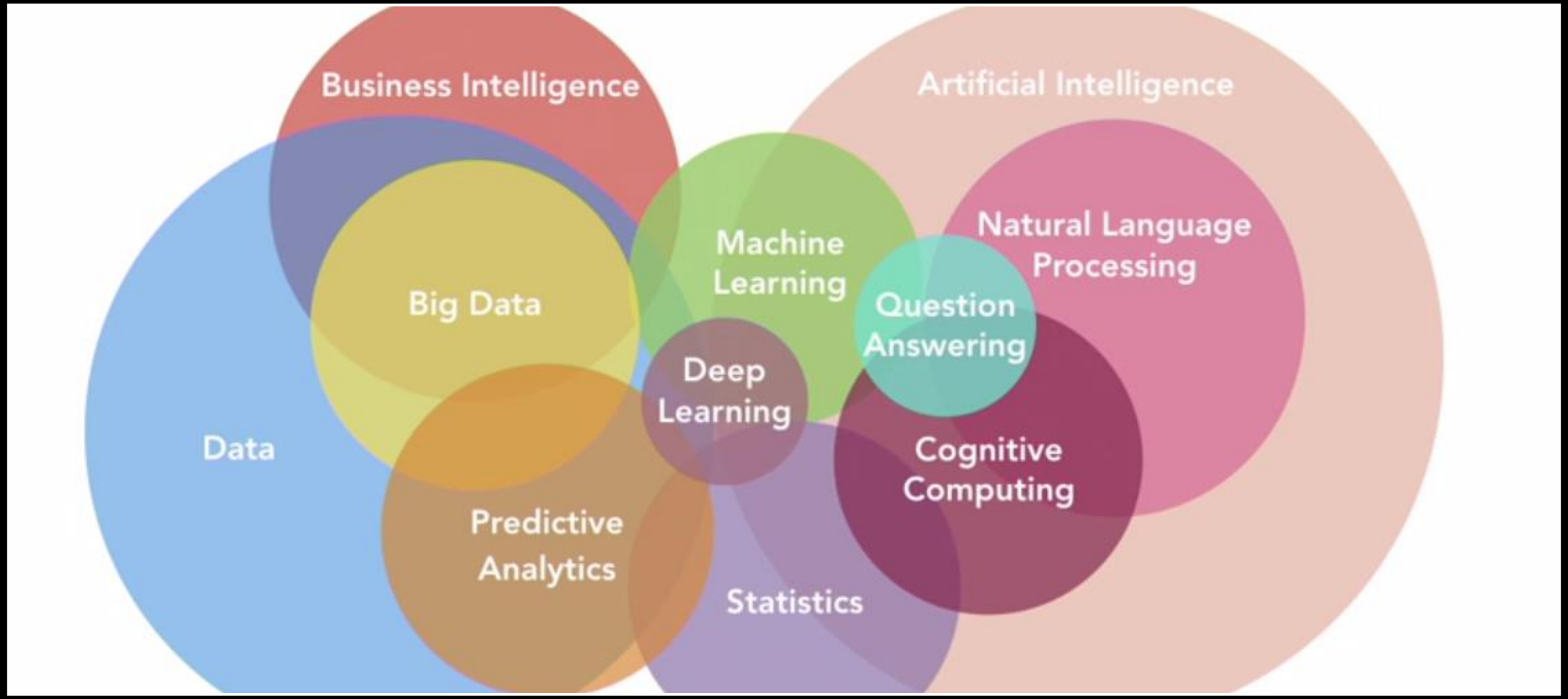
Out[6]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

Scikit-learn is a very popular open-source machine learning project which is constantly being developed and improved.

Why should you use scikit-learn?

- Provides a large number of machine learning algorithms
- Has a clean, uniform, and streamlined API
- Works well with other python libraries
- Switching algorithms is very straightforward
- Widely used in industry and academia

Scikit-learn is a great library for machine learning.

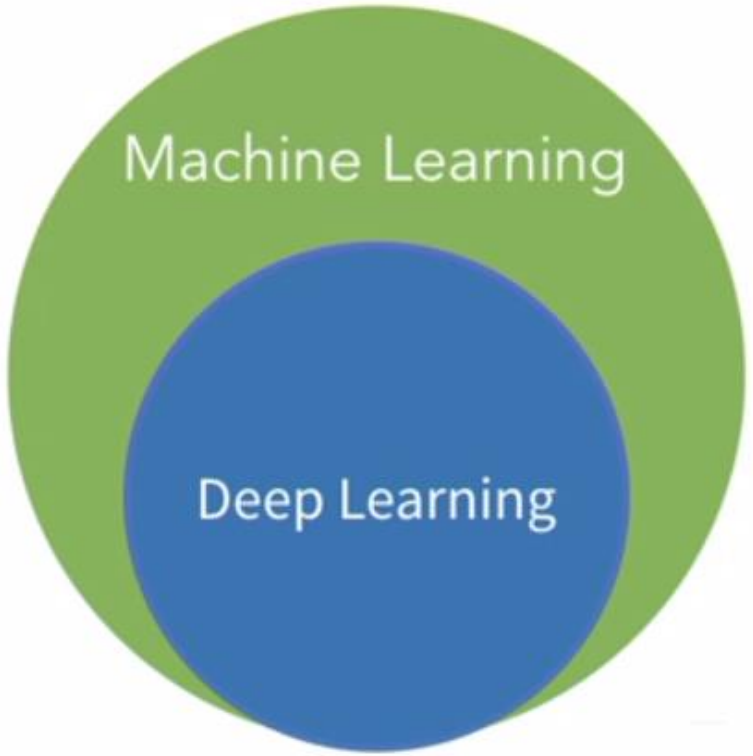


Machine Learning Vs Deep Learning Vs Artificial Intelligence !!!

Fitting a function to examples and using that function to generalize and make predictions about new examples

Alternative: pattern matching

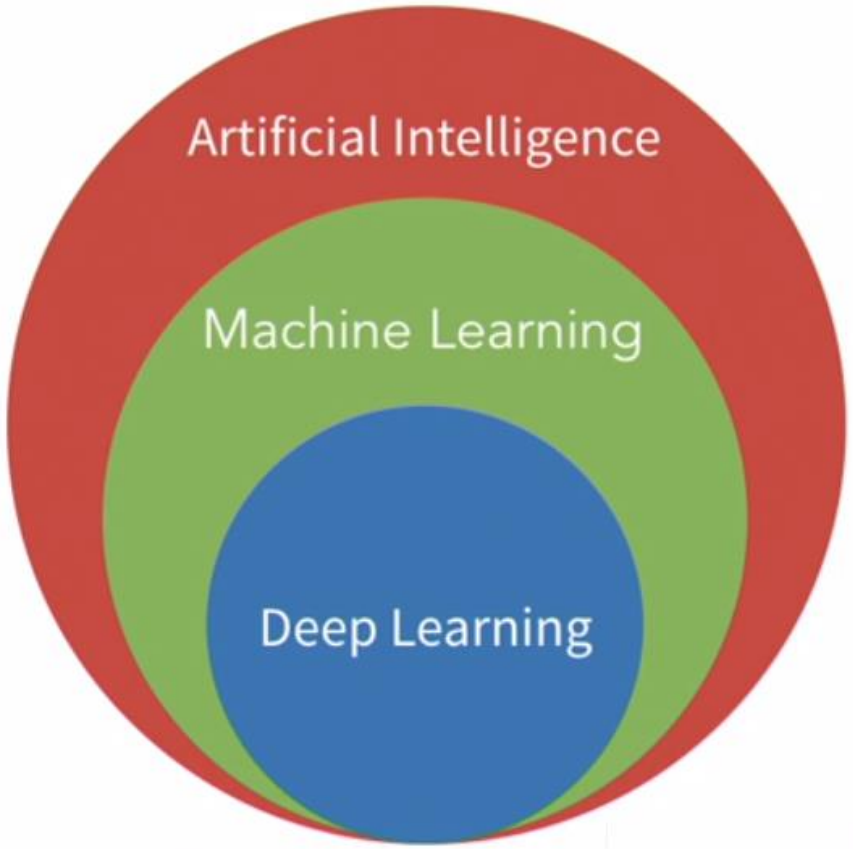
Fitting functions to examples where those functions are connected layers of nodes; with the intent to generalize and make predictions about new examples



Alternative: connected pattern matching

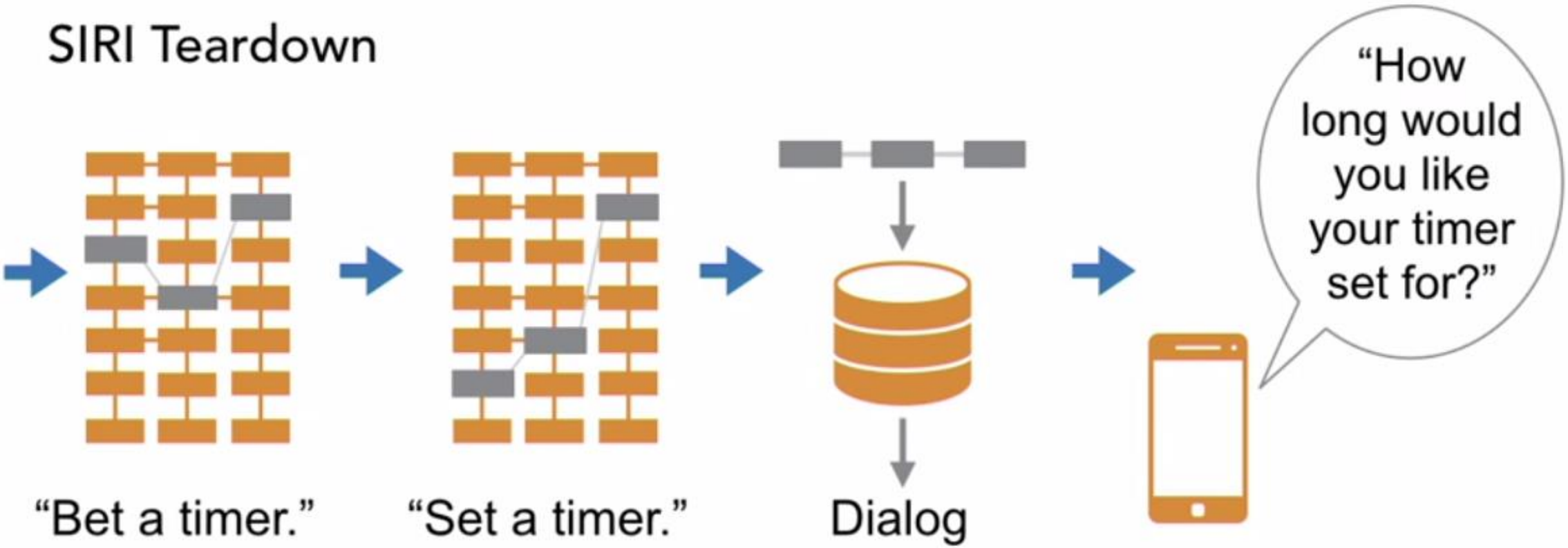
Weak AI – intelligence specifically designed to focus on a narrow task

Strong/general AI – a machine with consciousness, sentience, and a mind; general intelligence capable of any and all cognitive functions and reasoning that a human is capable of

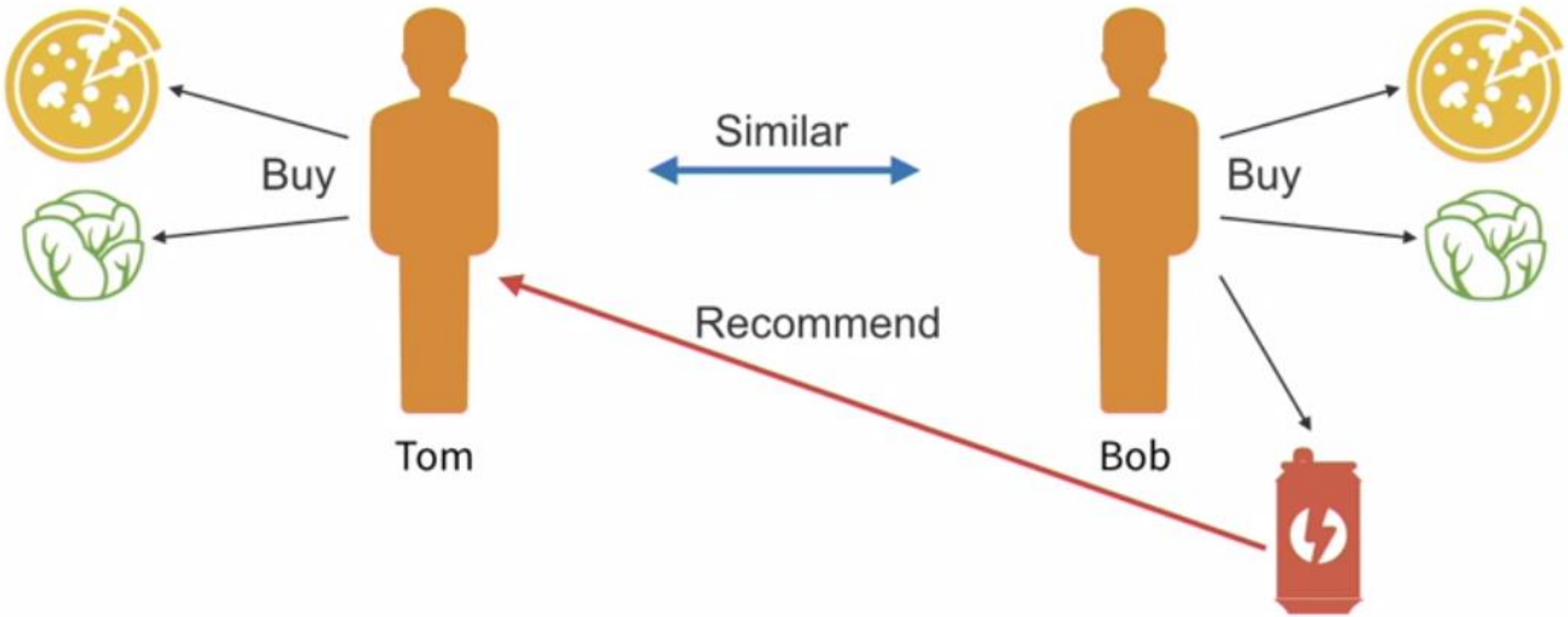


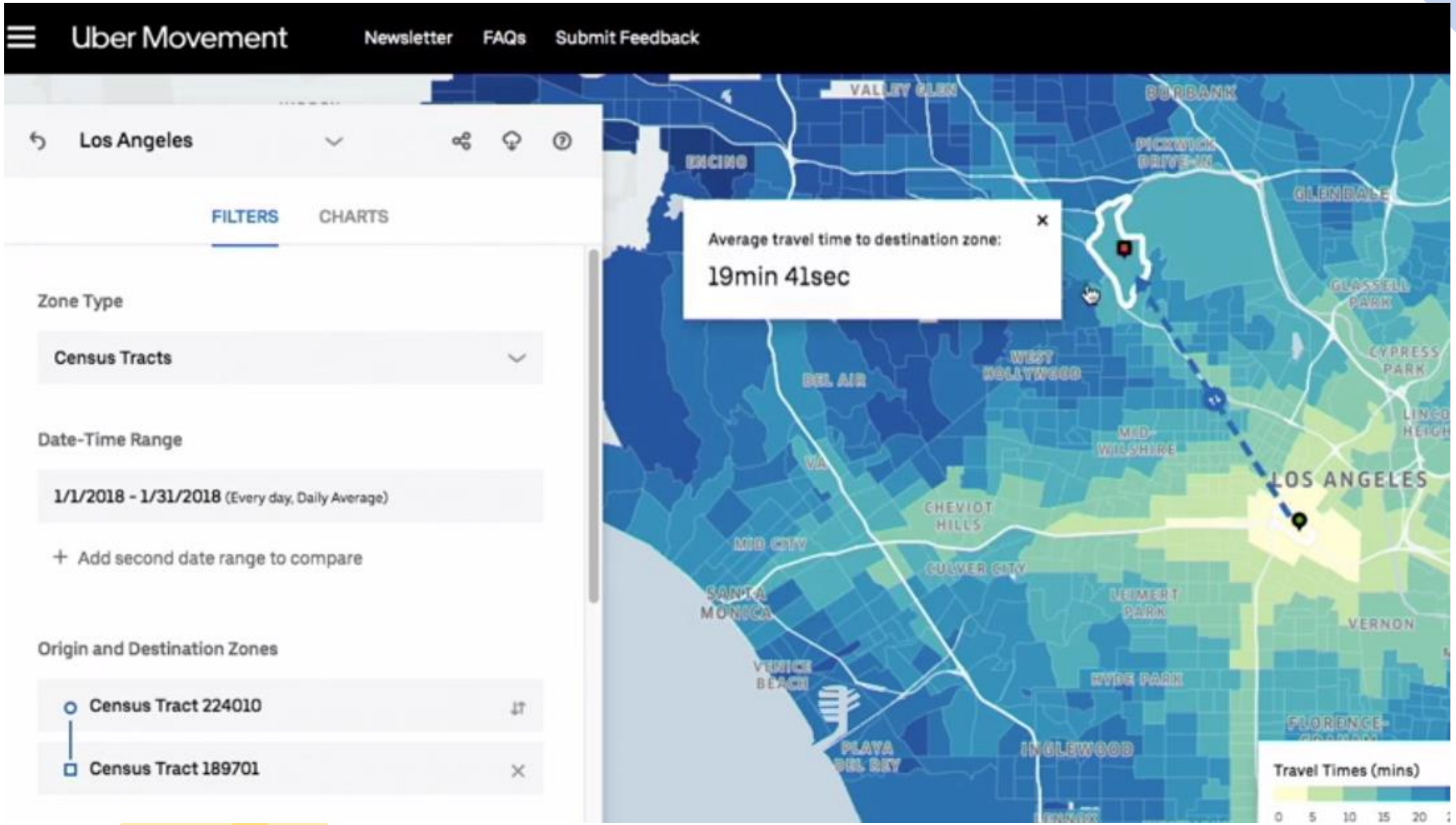
Personal assistants – Siri, Google Home, Amazon Echo

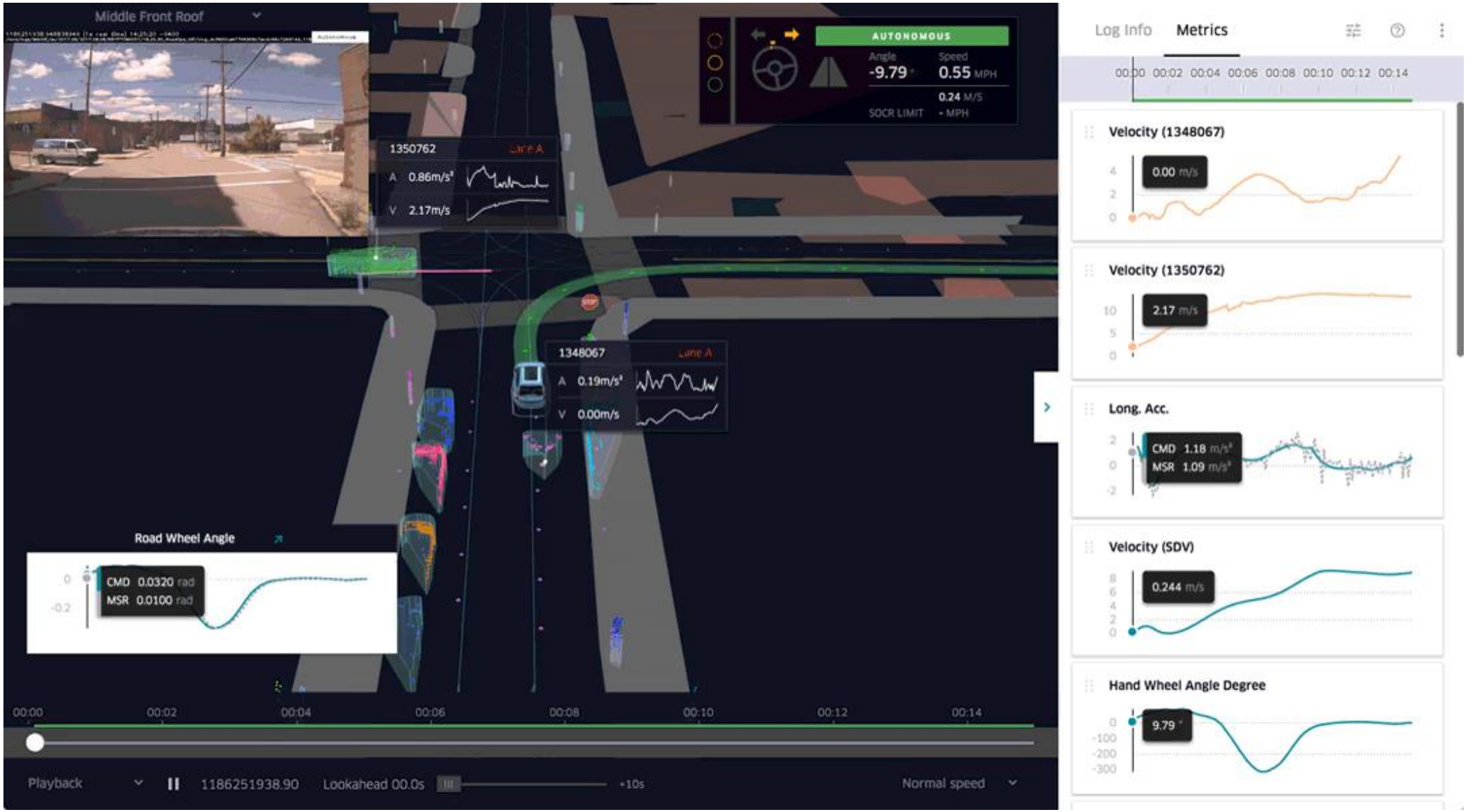
SIRI Teardown



Recommendation systems – Amazon, Netflix, Facebook, Twitter




Uber Movement
Newsletter
FAQs
Submit Feedback
Los Angeles
FILTERS
CHARTS
Zone Type
Census Tracts
Date-Time Range
1/1/2018 - 1/31/2018 (Every day, Daily Average)
+ Add second date range to compare
Origin and Destination Zones
Census Tract 224010
Census Tract 189701
Average travel time to destination zone:
19min 41sec
LOS ANGELES
Travel Times (mins)
0 5 10 15 20


Middle Front Roof
AUTONOMOUS
Angle
-9.79
Speed
0.55 MPH
0.24 M/S
SOCR LIMIT
- MPH
1350762
Lane A
A 0.86m/s²
V 2.17m/s
1348067
Lane A
A 0.19m/s²
V 0.00m/s
Road Wheel Angle
CMD 0.0320 rad
MSR 0.0100 rad
00:00
00:02
00:04
00:06
00:08
00:10
00:12
00:14
Playback
1186251938.90
Lookahead 00.0s
+10s
Normal speed
Log Info
Metrics
Velocity (1348067)
0.00 m/s
Velocity (1350762)
2.17 m/s
Long. Acc.
CMD 1.18 m/s²
MSR 1.09 m/s²
Velocity (SDV)
0.244 m/s
Hand Wheel Angle Degree
9.79 °

Assess the Problem You're Trying to Solve

Is this a type of problem that can be solved using machine learning?

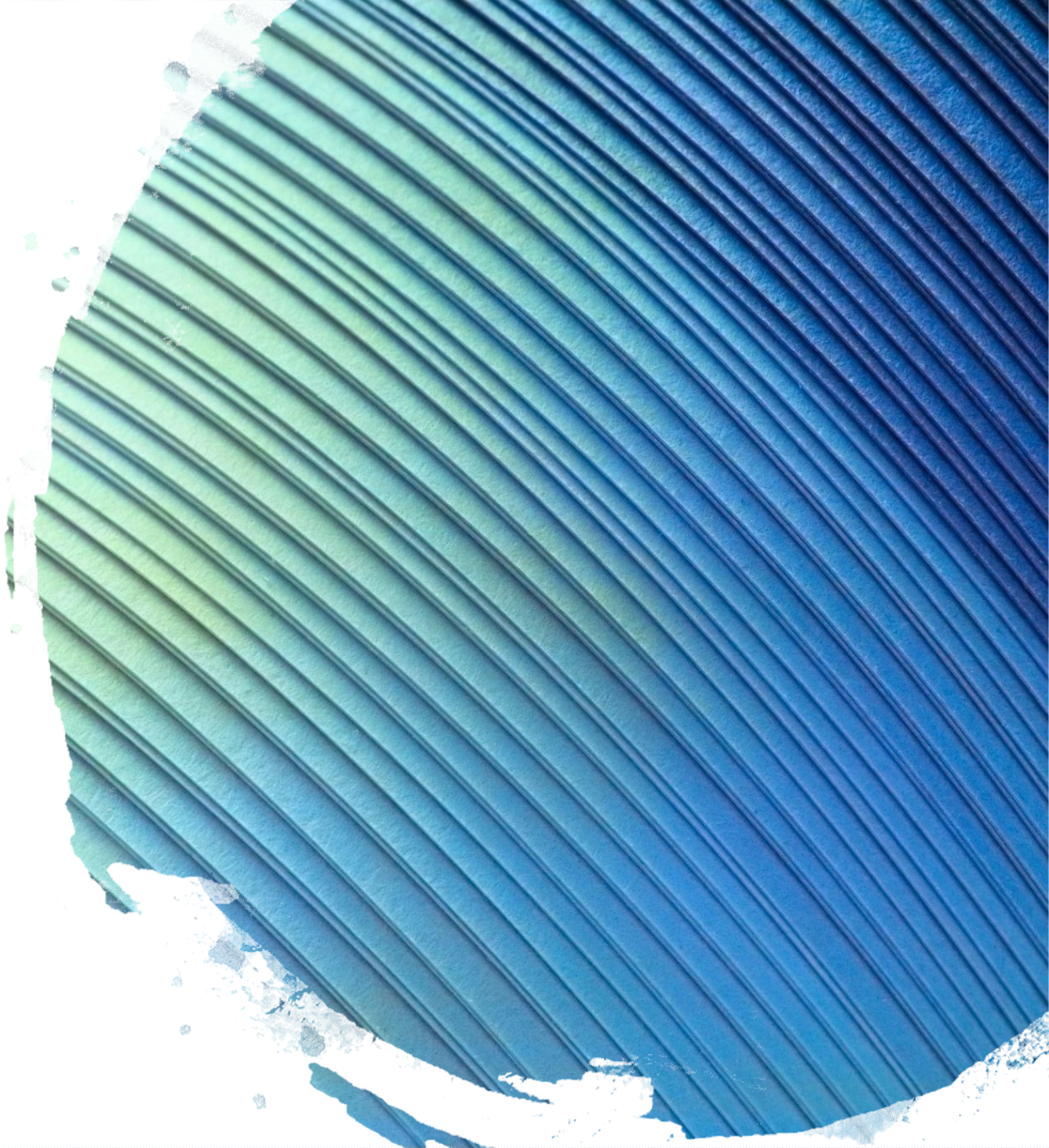
- Does it require a prediction or bucketing something into categories?
- Is the problem relatively self-contained?

Is my particular problem one that can be solved with machine learning?

- Do I have data with labels?
- Do I have the ability to assess the quality of the model?
- Can I determine an acceptable accuracy threshold?

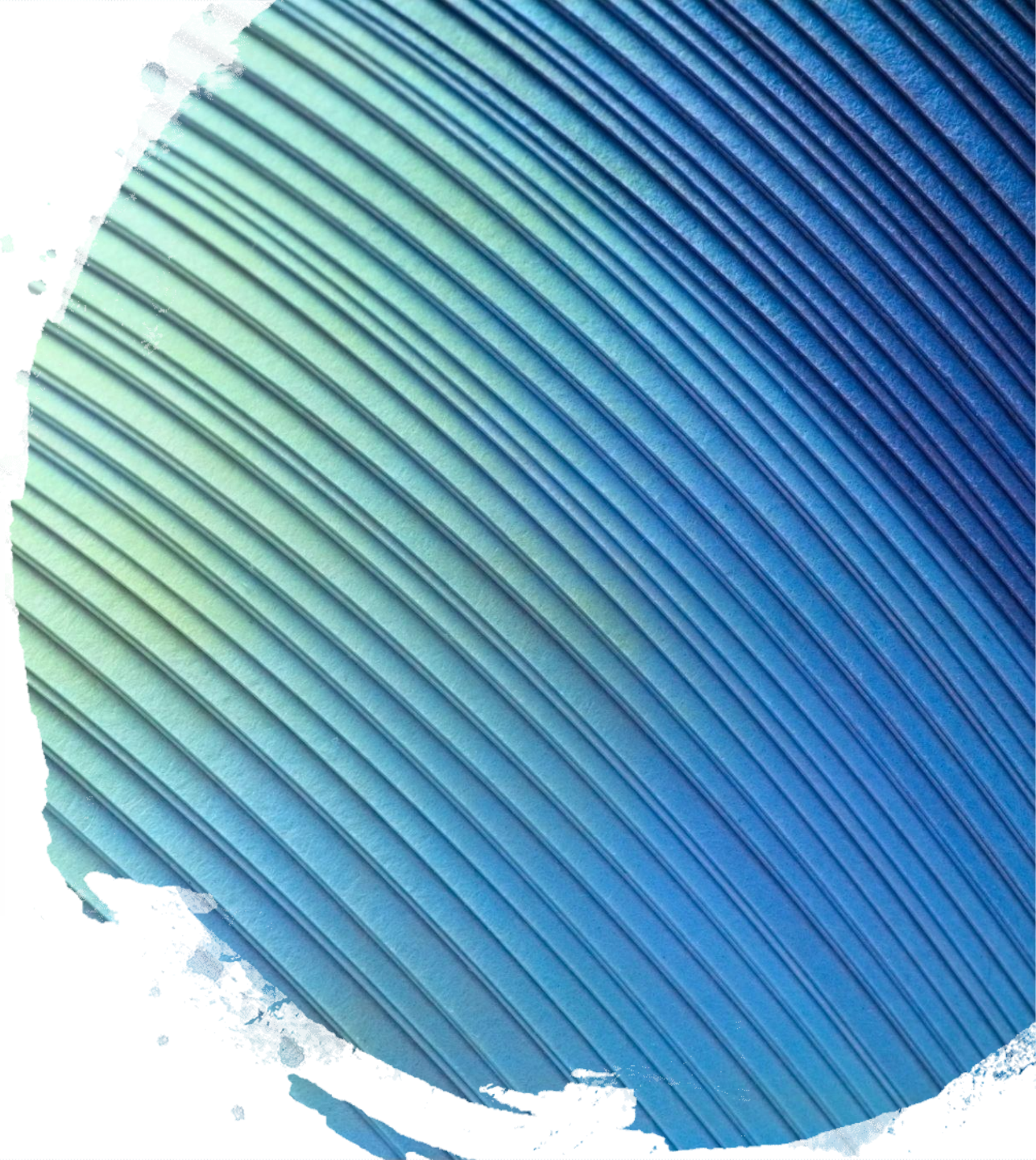
Common Challenges!!!

- Problem Scoping
- Data
- Infrastructure
- Latency



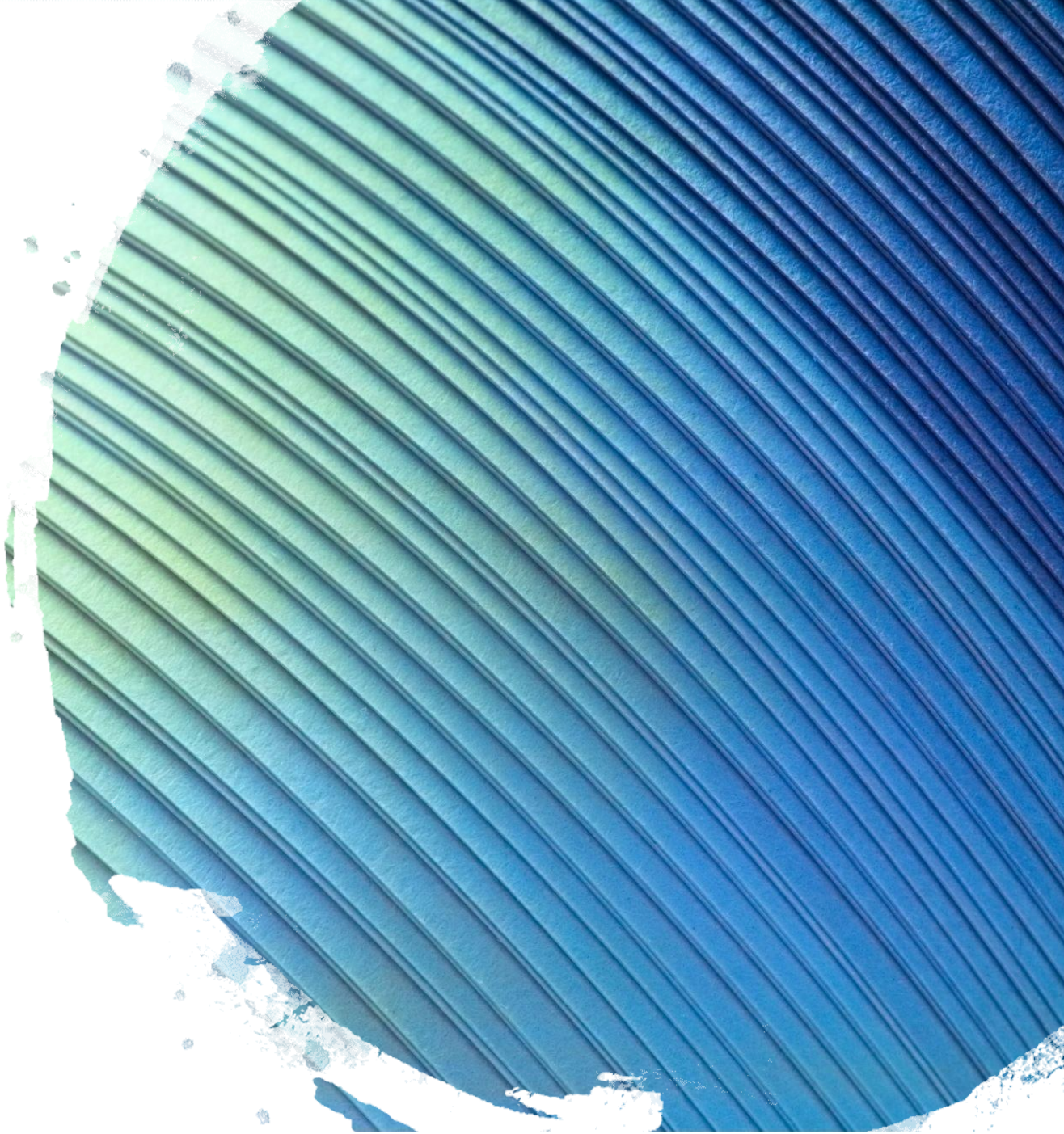
Problem scoping

- The right problem isn't being solved
- Tolerance threshold is undetermined
- Impact in real environment can't be measured



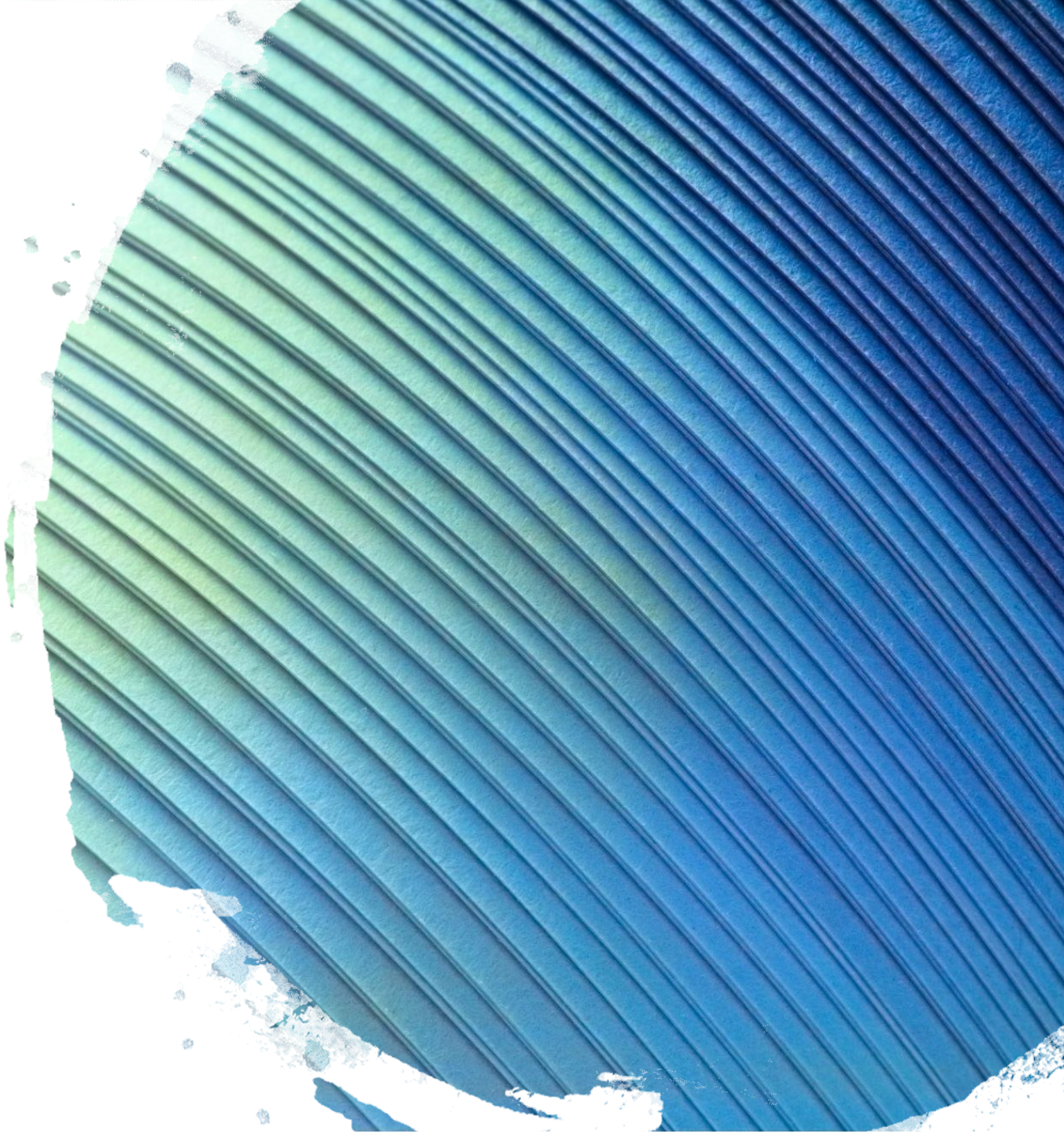
Data

- Lack of data/data sensitivity
- Too much data
- Data doesn't have labels



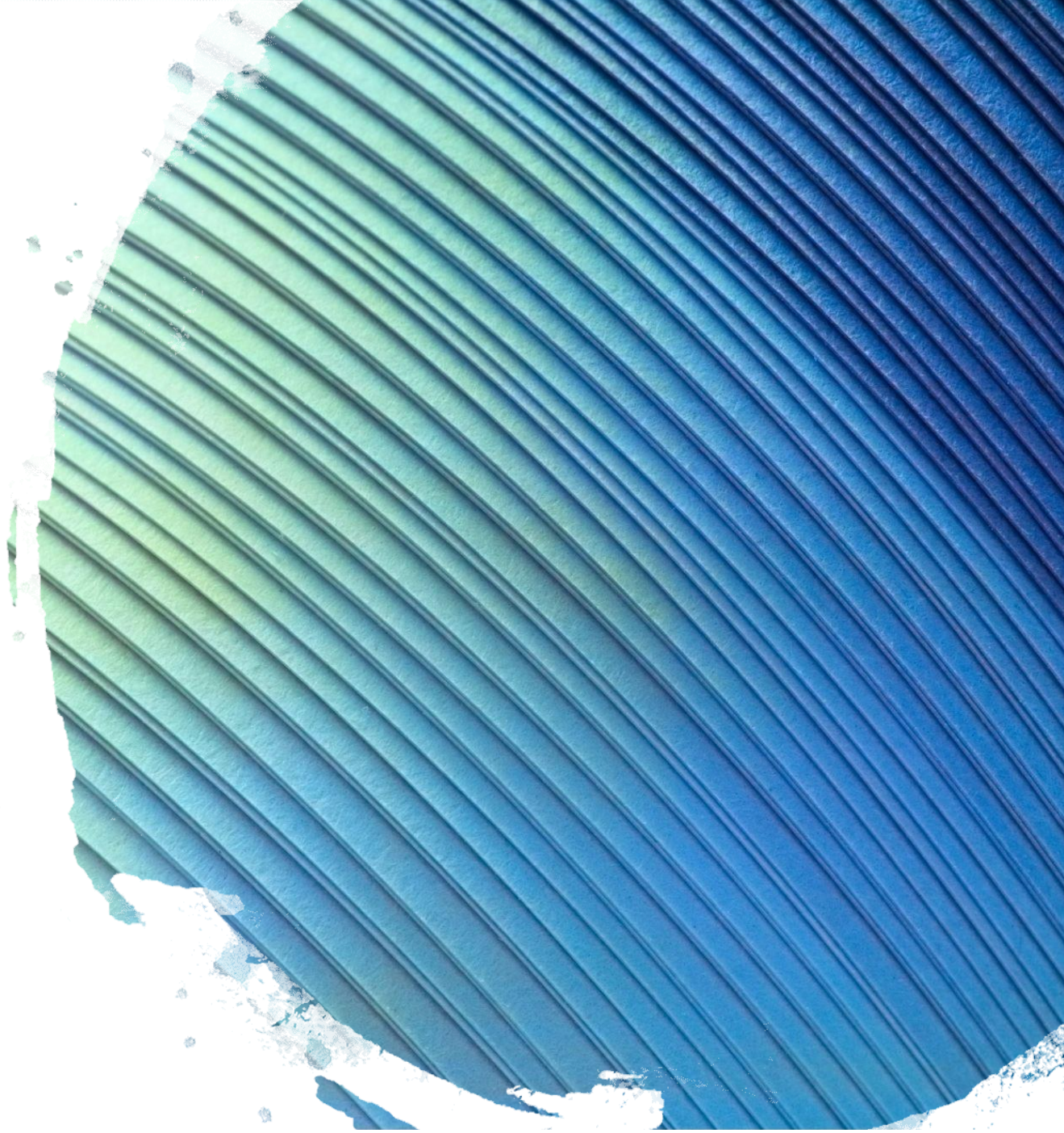
Data

- Lack of data/data sensitivity
- Too much data
- Data doesn't have labels
- Data is too biased, dirty, noisy, incomplete, etc.



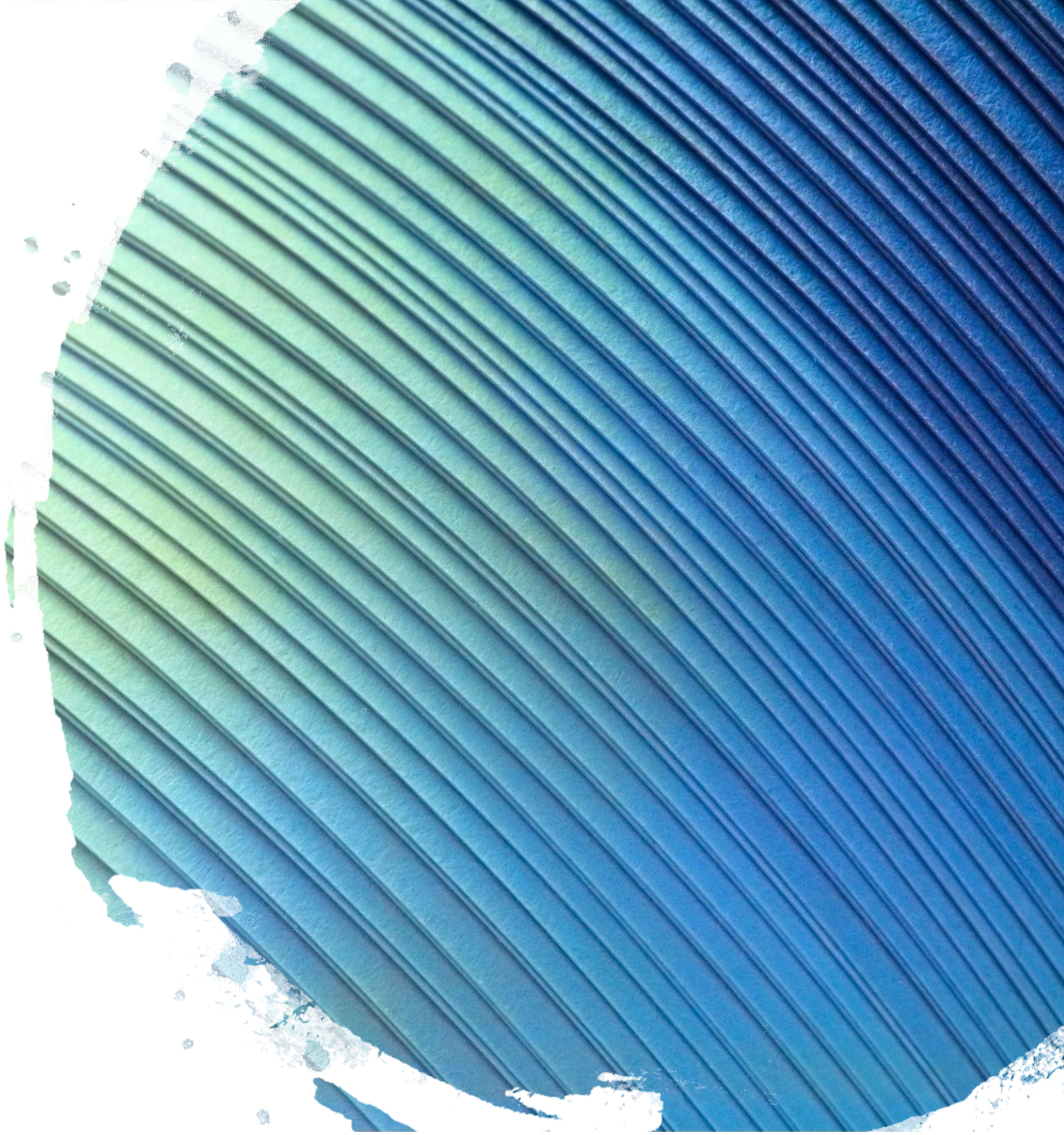
Infrastructure

- Lack skills to robustly automate
- Not enough compute power
- Inability to A/B test with existing solution
- Inability to continuously track model quality



Latency

- Model takes too long to train
- Model takes too long at inference time



Exploratory Data Analysis

Why?

- Understand the shape of the data
- Learn which features might be useful
- Inform the cleaning that will come next

What?

- Counts or distributions of all variables
- Data type for each feature
- Missing data
- Correlations
- Duplicates

Data Cleaning

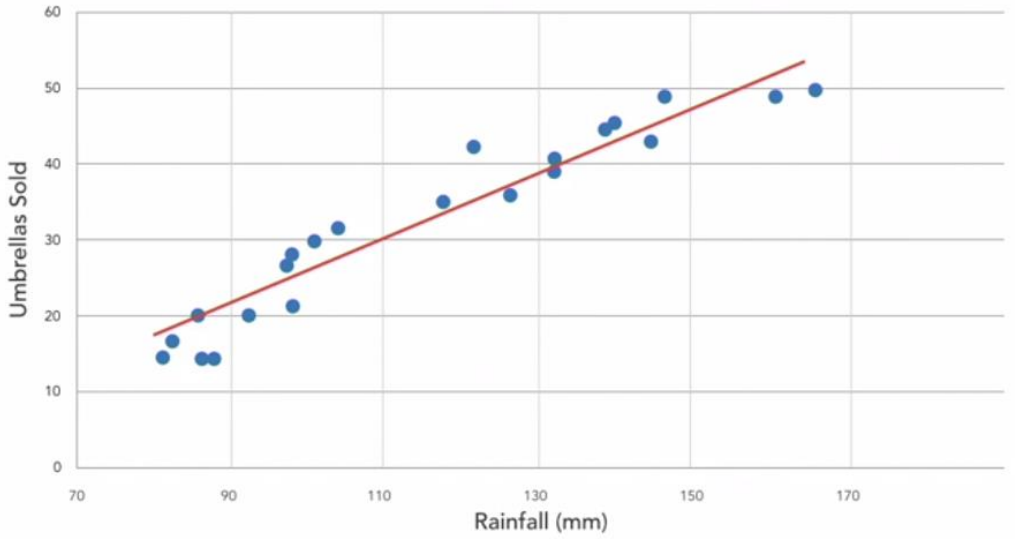
Why?

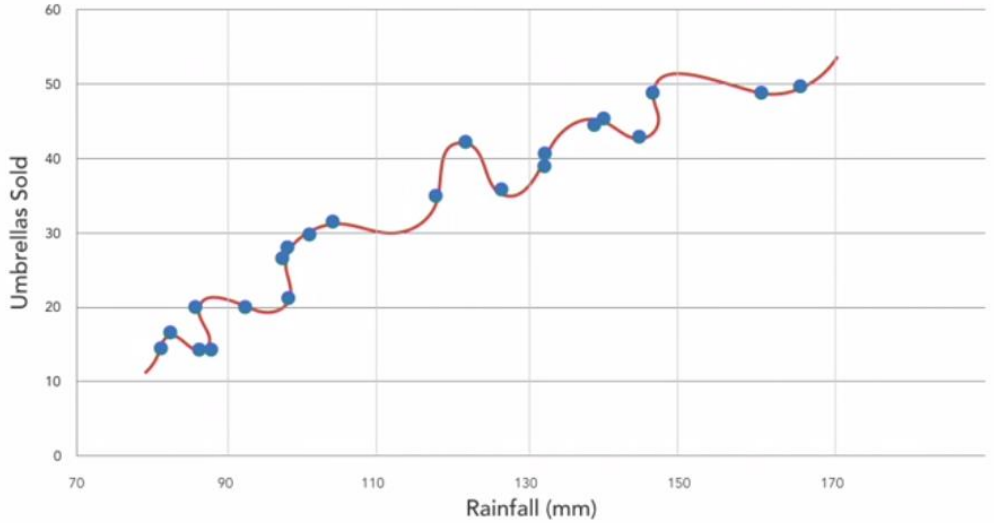
- Shape data in a way a model can best pick up on the signal
- Remove irrelevant data
- Adjust features to be acceptable for a model

What?

- Anonymize
- Encode categorical variables
- Fill missing data (if necessary)

Fitting a function to examples and using that function to generalize and make predictions about new examples

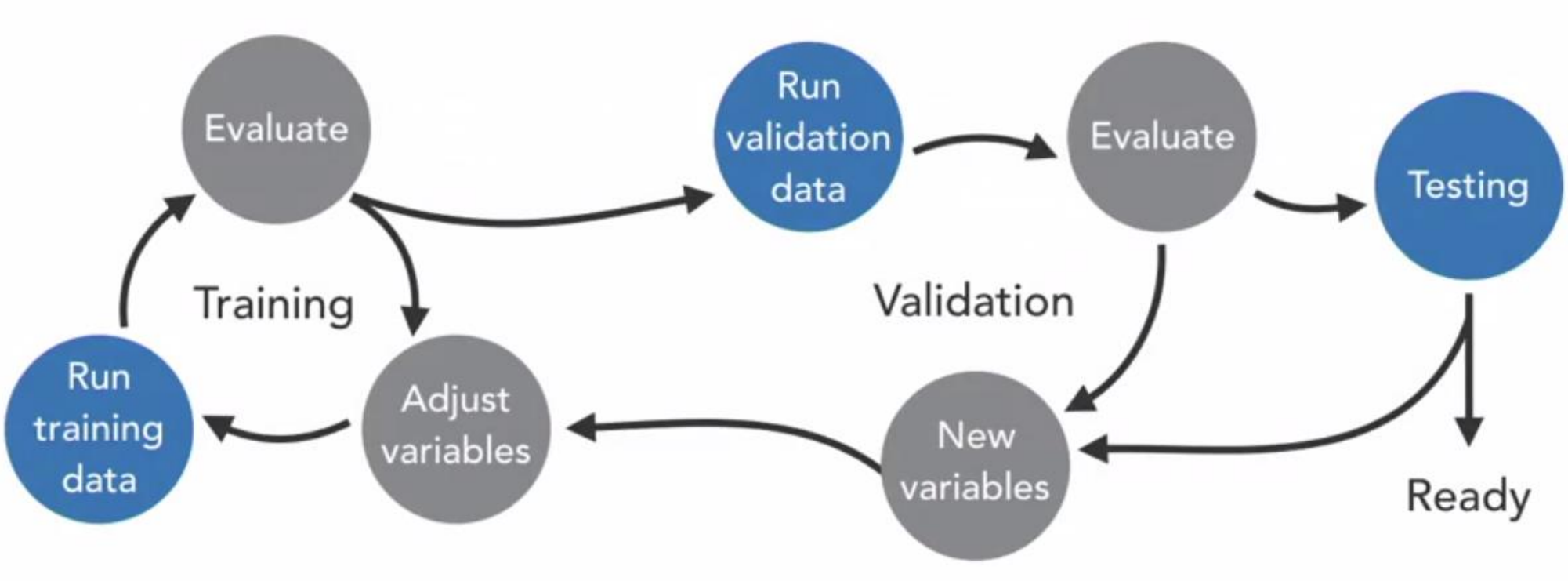




Will This Generalize?

How Do We Split Up Our Data?

- Training dataset – data used to train the model (allow the algorithm to learn from this data)
- Validation dataset – data used to select the best model (optimal algorithm and hyperparameter settings)
- Test dataset – data used to provide an unbiased evaluation of what the model will look like in its real environment


Evaluate
Run validation data
Evaluate
Testing
Training
Validation
Run training data
Adjust variables
New variables
Ready

What if we don’t split the dataset?

- Overfitting or underfitting to the data
- Inaccurate representation of how the model will generalize

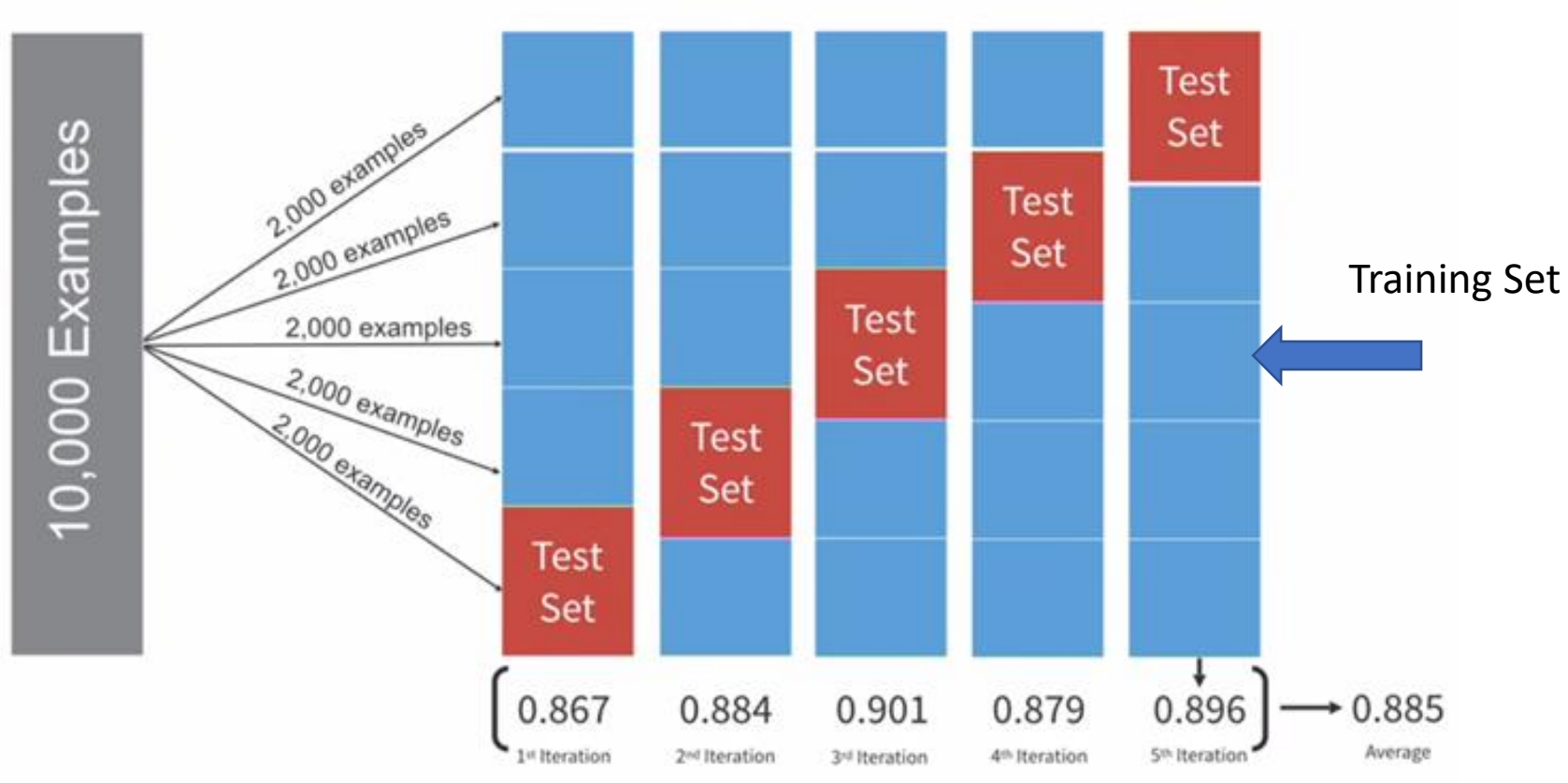
Holdout Test Set

Sample of data not used in fitting a model; used to evaluate the model's ability to generalize to unseen data

K-Fold Cross Validation

Data is divided into k subsets and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are combined to be used to train the model.

K-Fold Cross Validation Process

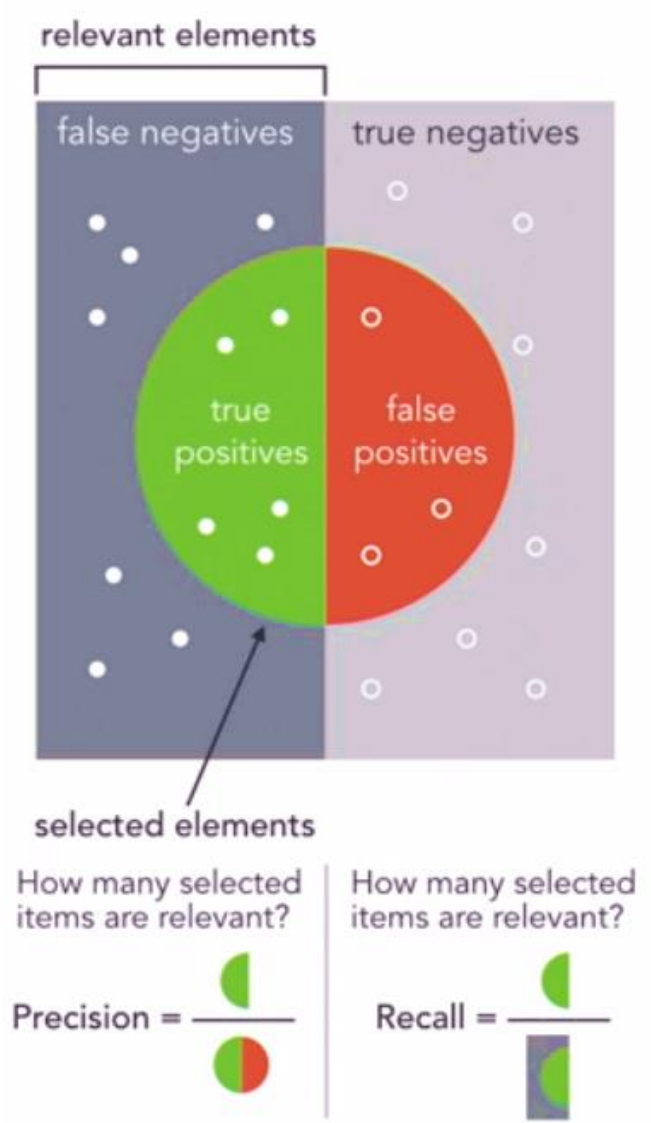


Evaluation Framework

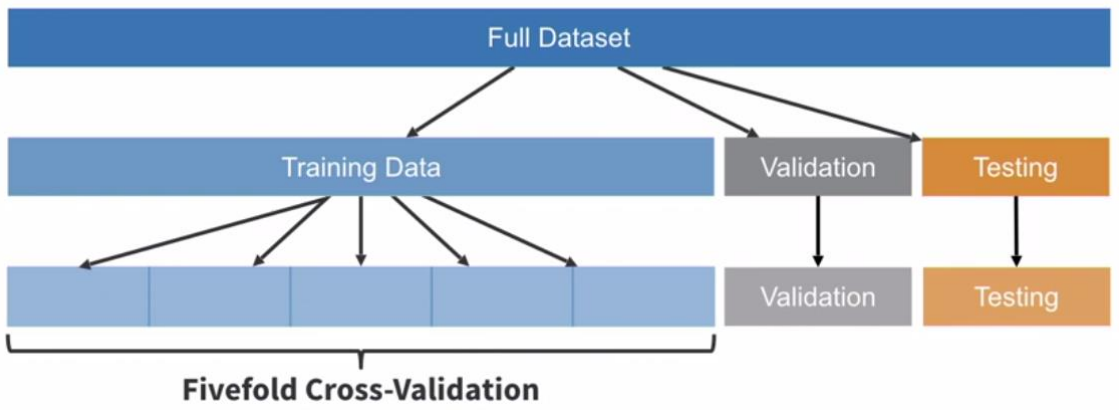
1. Evaluation metrics – how are we gauging the accuracy of the model?
2. Process (how to split the data) – how do we leverage our full dataset to mitigate likelihood of overfitting or underfitting?

Evaluation Metrics

- Accuracy = $\frac{\text{\# predicted correctly}}{\text{total \# of examples}}$
- Precision = $\frac{\text{\# predicted as surviving that actually survived}}{\text{total \# predicted to survive}}$
- Recall = $\frac{\text{\# predicted as surviving that actually survived}}{\text{total \# that actually survived}}$

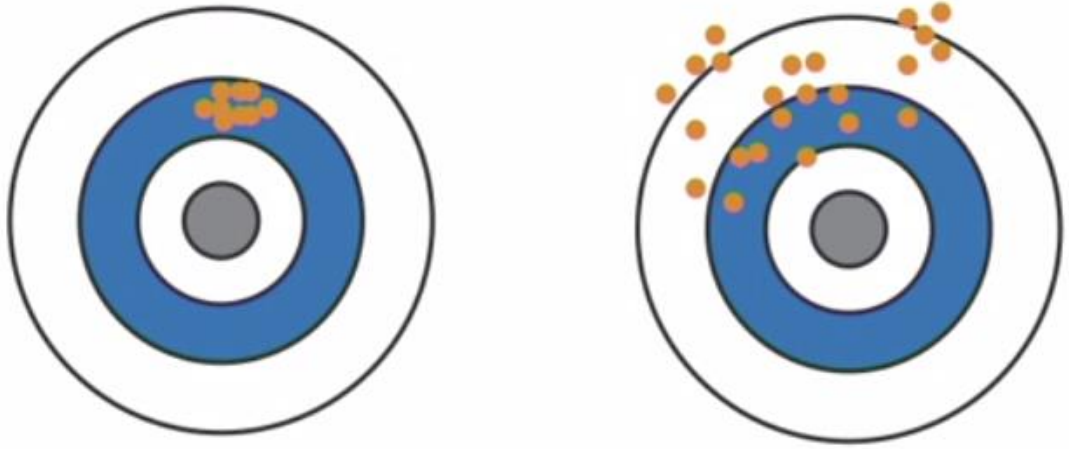


Process



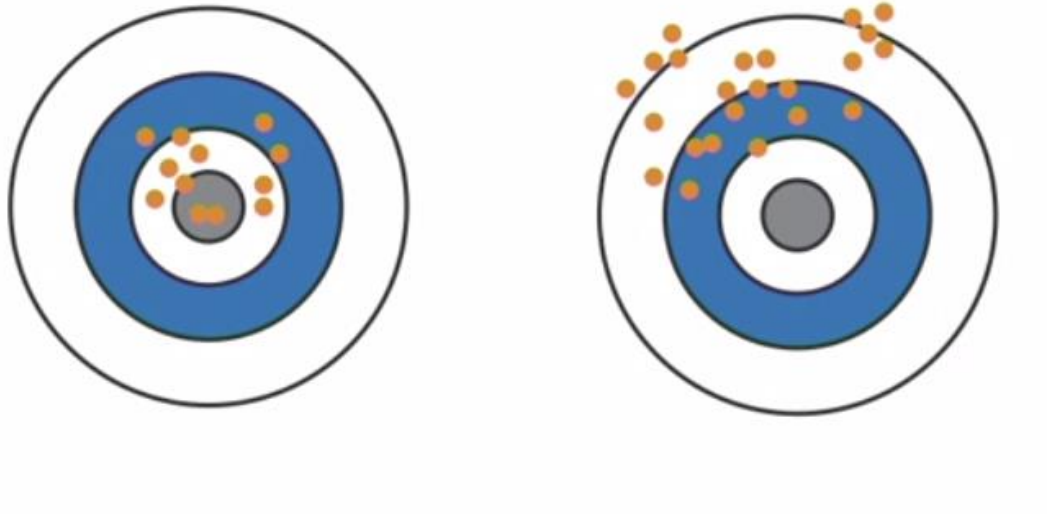
1. Run fivefold cross-validation and select the best models.
2. Re-fit models on full training set, evaluate those models on the validation set and pick the best one.
3. Evaluate that best model on the test set to gauge its ability to generalize to unseen data.


Low Variance
High Variance
Low Bias
High Bias



Bias is the algorithm's tendency to consistently learn the wrong thing by not taking into account all the information in the data.

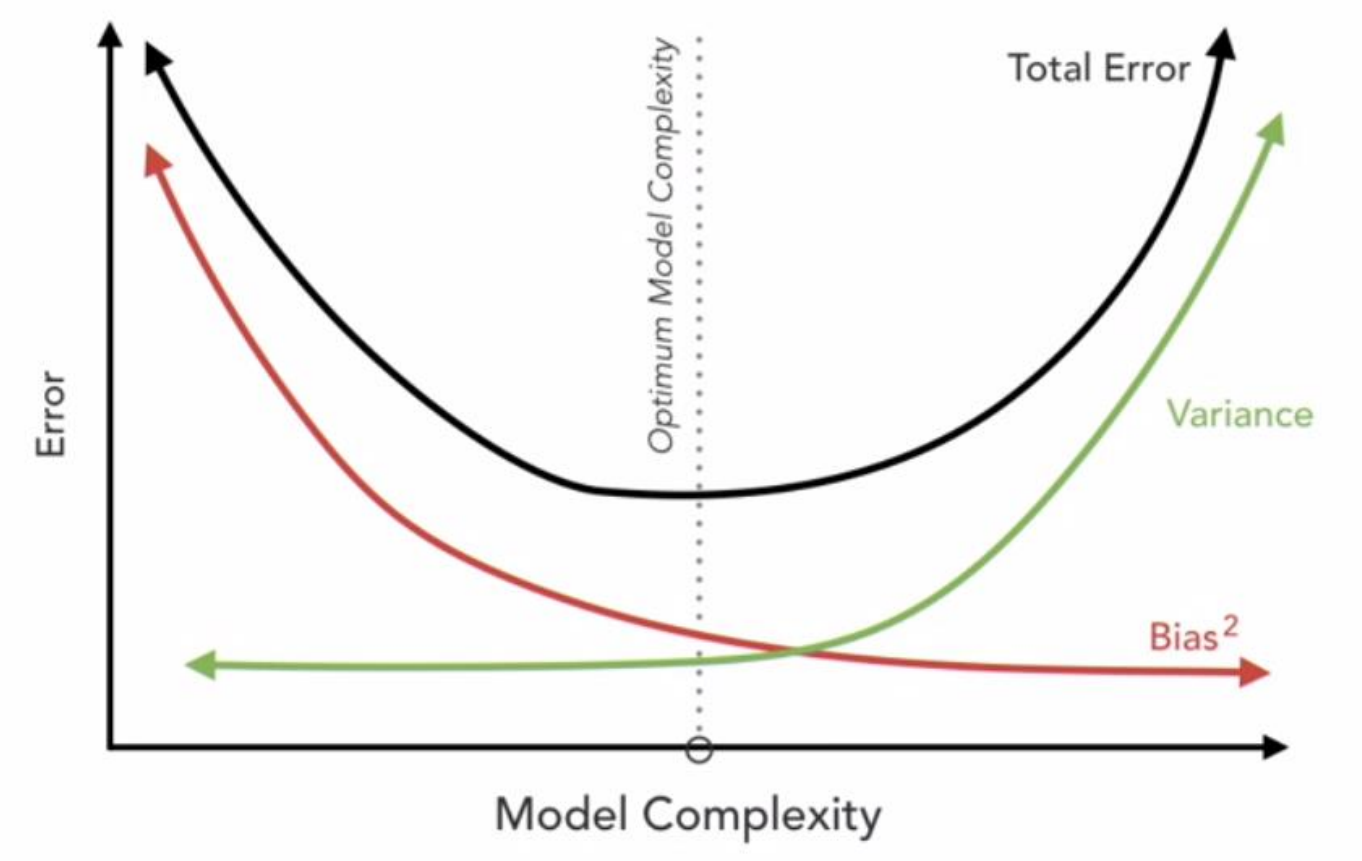
High bias is a result of the algorithm missing the relevant relations between features and target outputs.



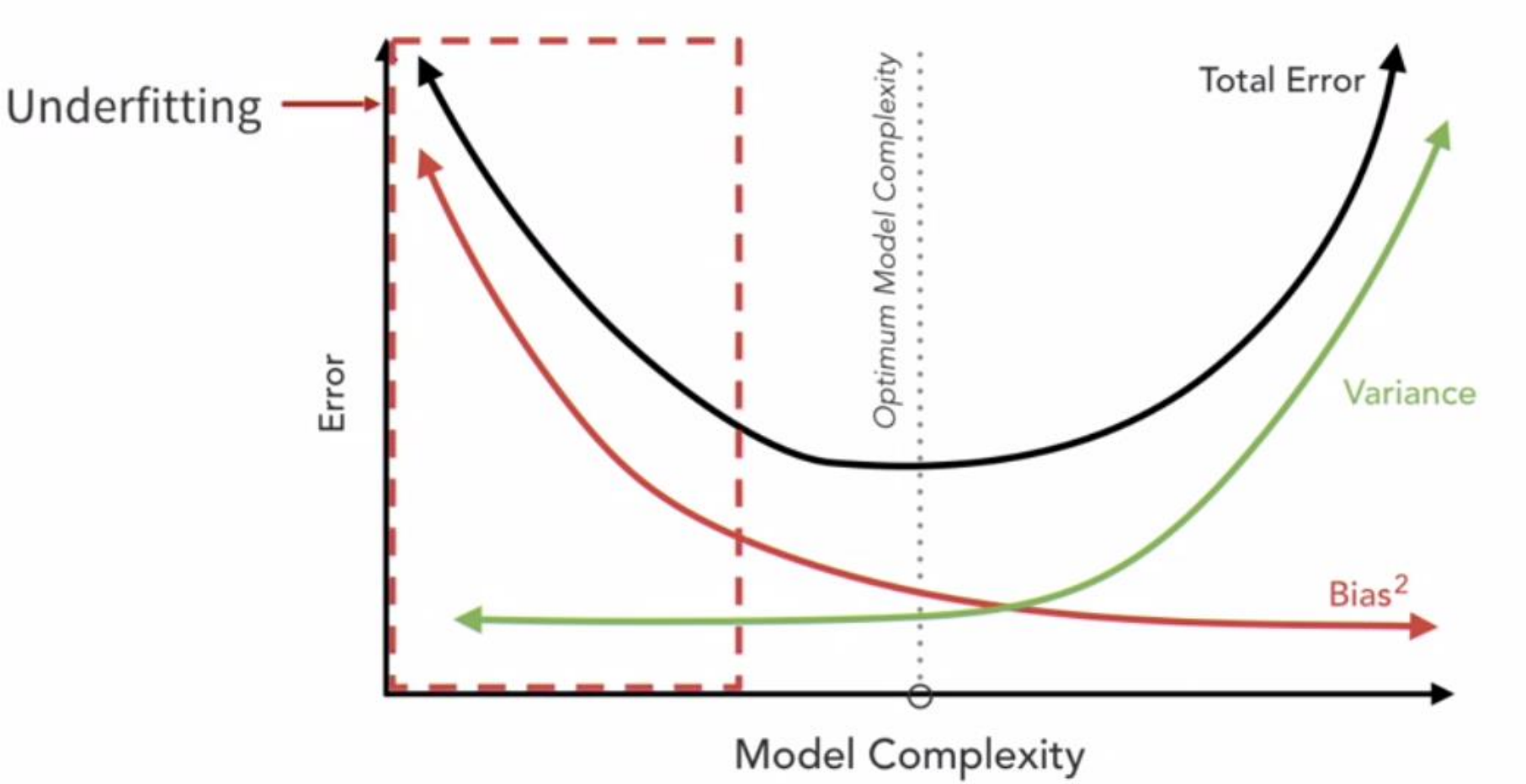
Variance refers to an algorithm's sensitivity to small fluctuations in the training set.

High variance is a result of the algorithm fitting to random noise in the training data.

Total Error = (Bias + Variance) + Irreducible Error



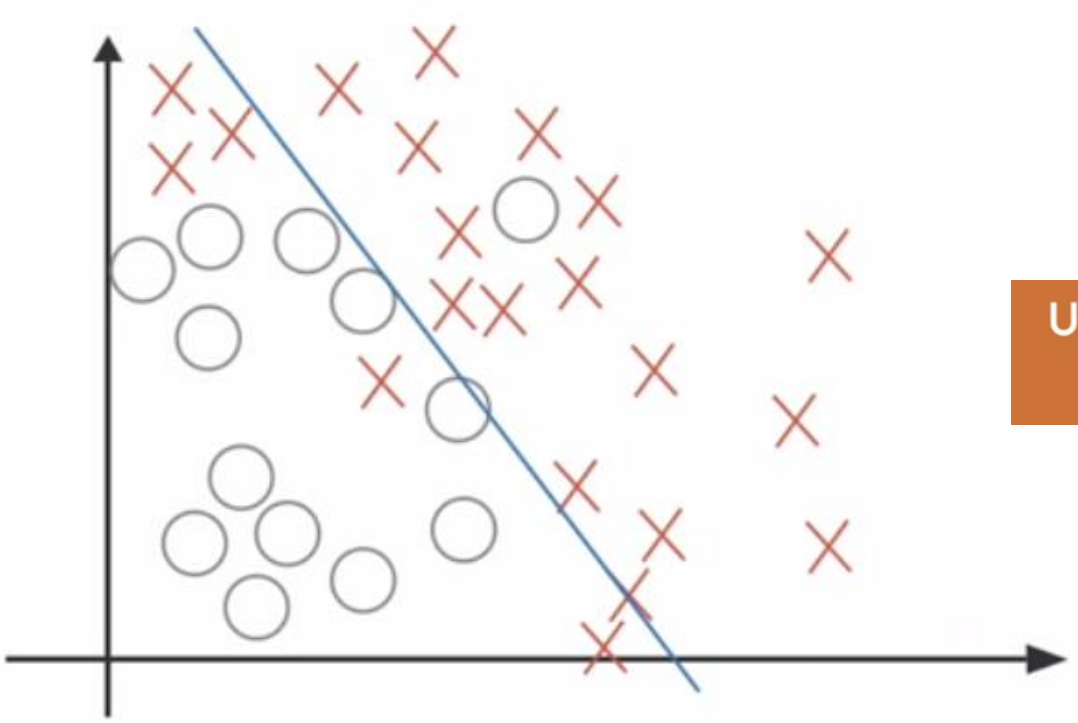
Total Error = (Bias + Variance) + Irreducible Error



Bias is the algorithm's tendency to consistently learn the wrong thing by not taking into account all the information in the data.

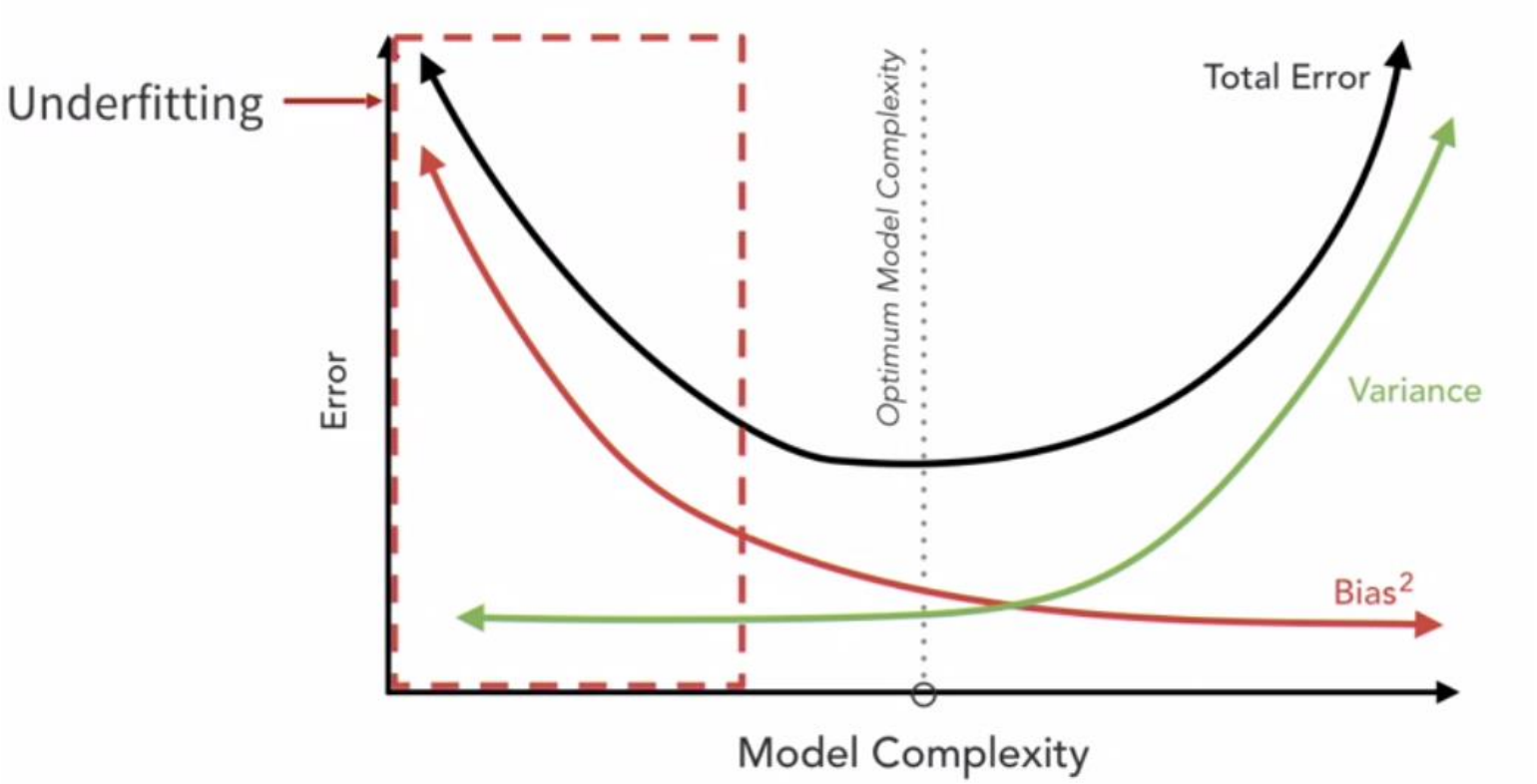
High bias is a result of the algorithm missing the relevant relations between features and target outputs.

Underfitting occurs when an algorithm cannot capture the underlying trend of the data.



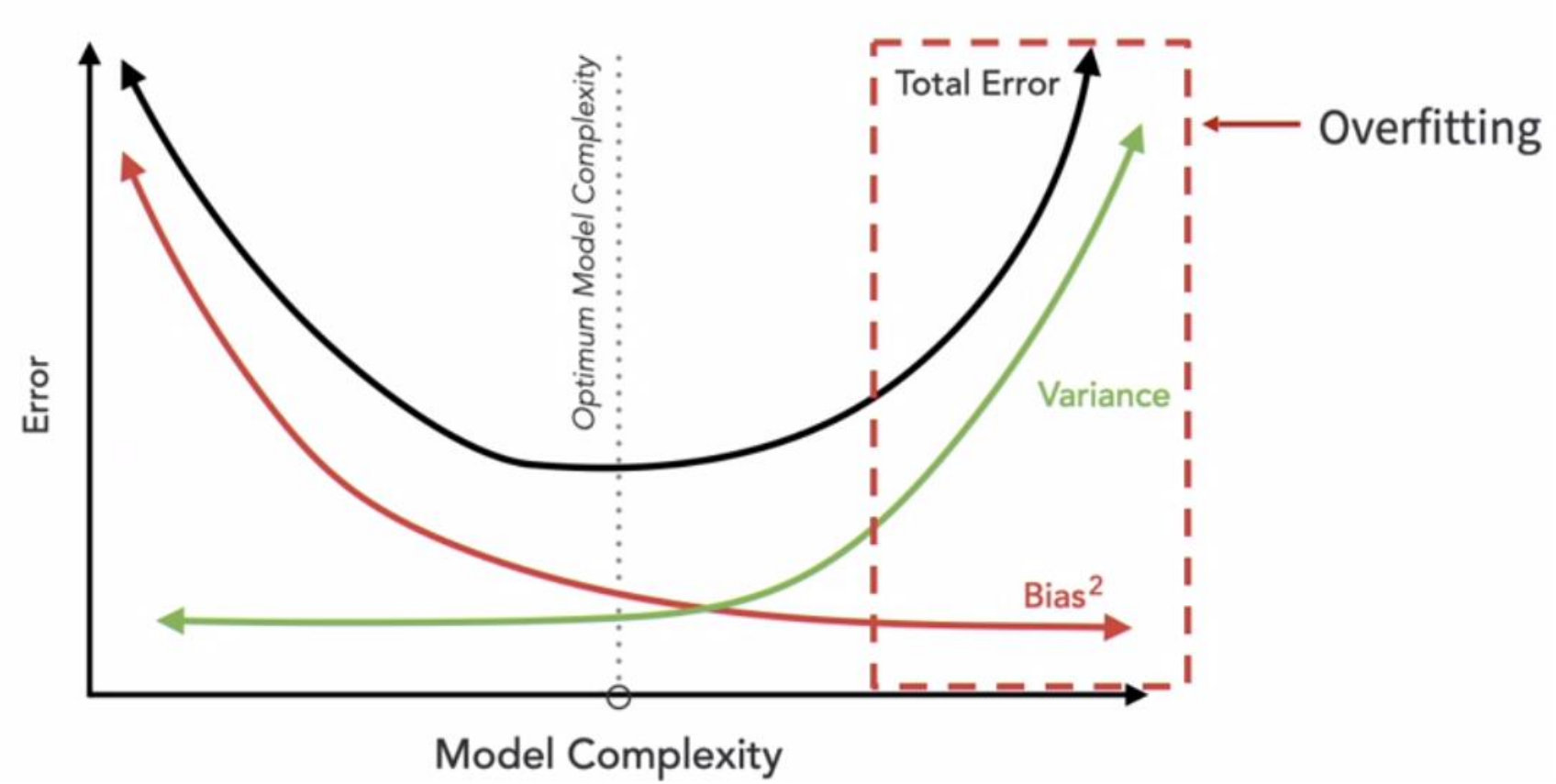
Underfitting occurs when an algorithm cannot capture the underlying trend of the data.

Total Error = (Bias + Variance) + Irreducible Error



Underfitting occurs when an algorithm cannot capture the underlying trend of the data.

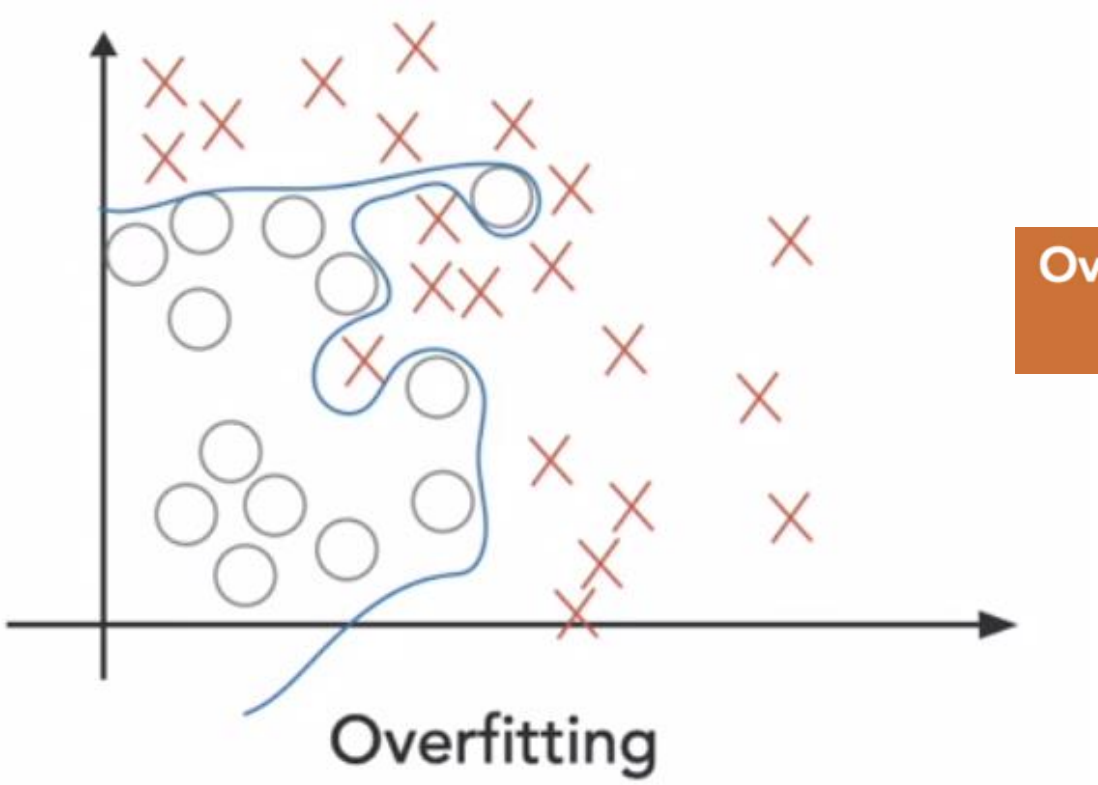
Total Error = (Bias + Variance) + Irreducible Error



Variance refers to an algorithm's sensitivity to small fluctuations in the training set.

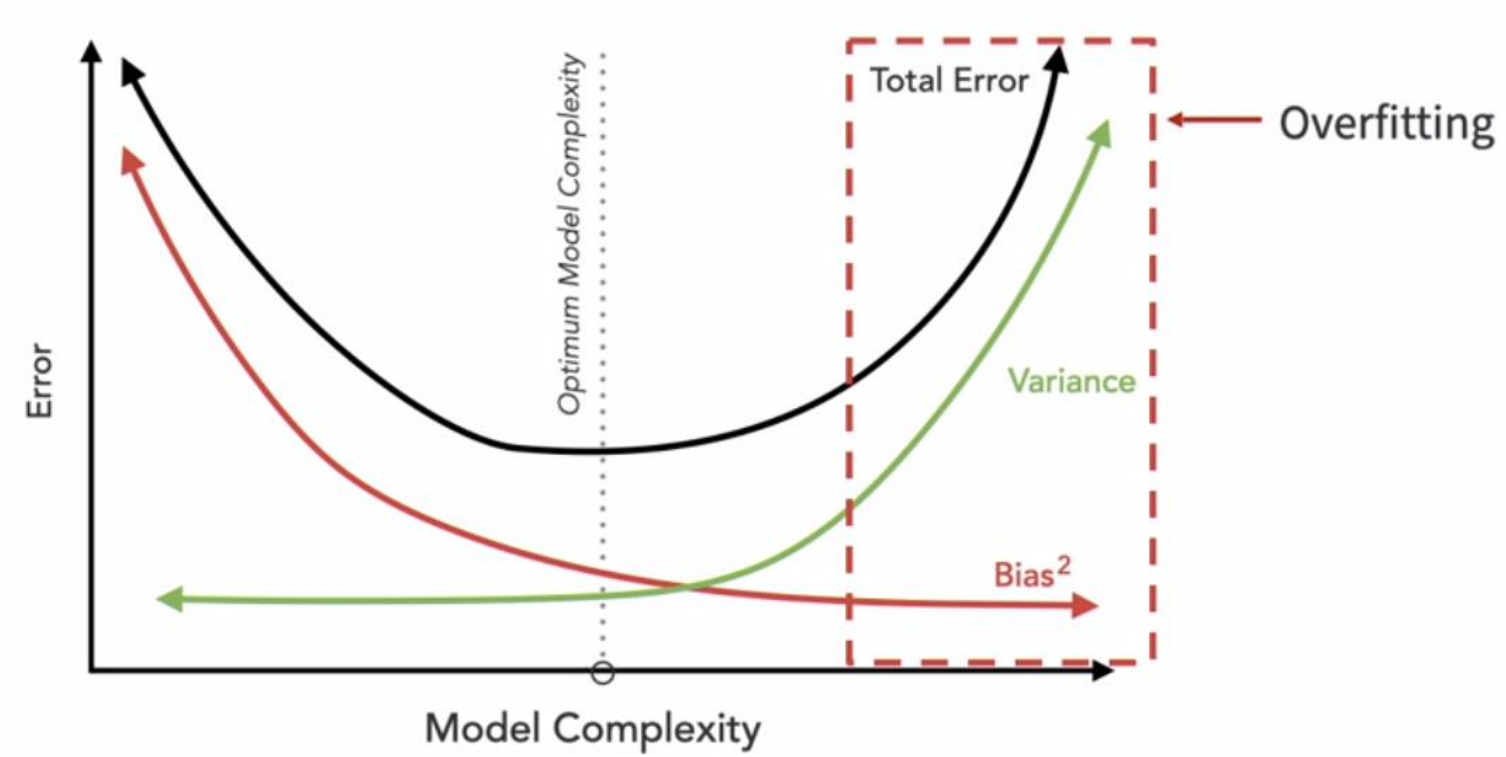
High variance is a result of the algorithm fitting to random noise in the training data.

Overfitting occurs when an algorithm fits too closely to a limited set of data.



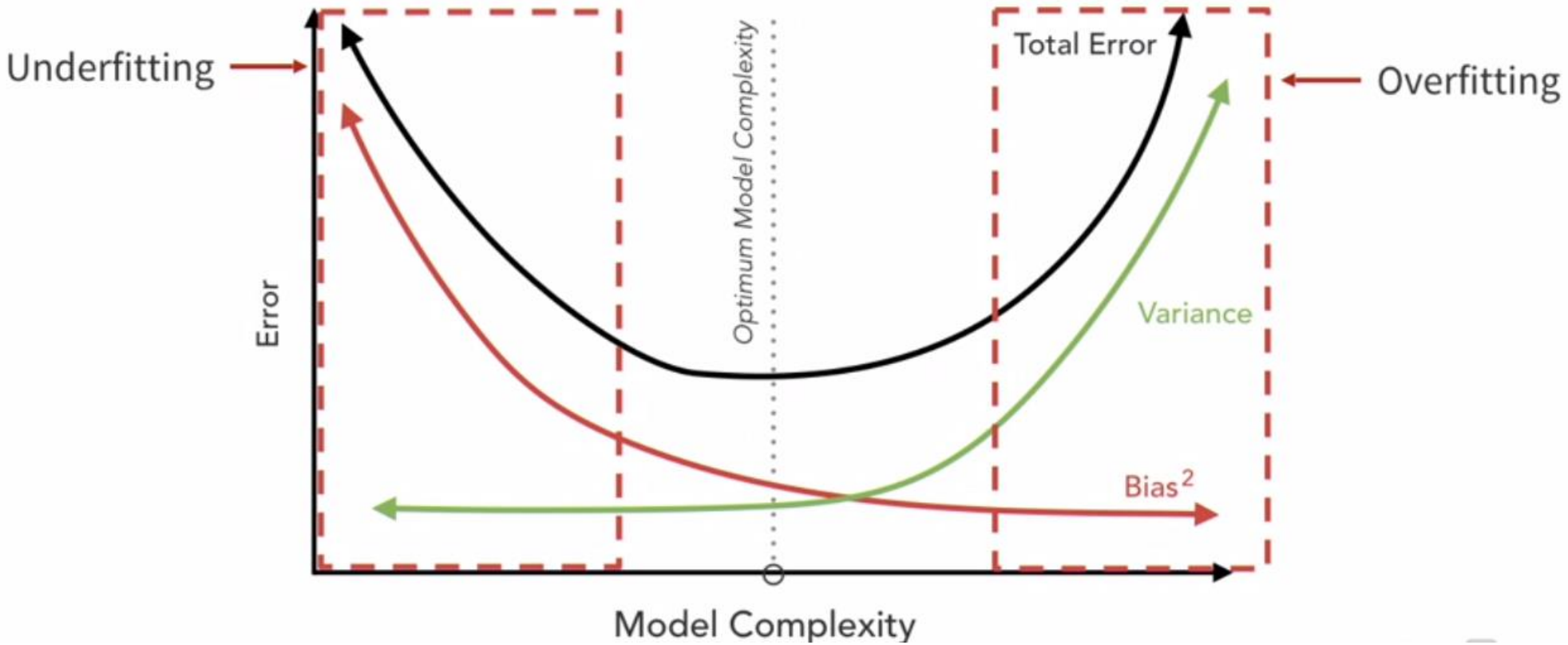
Overfitting occurs when an algorithm fits too closely to a limited set of data.

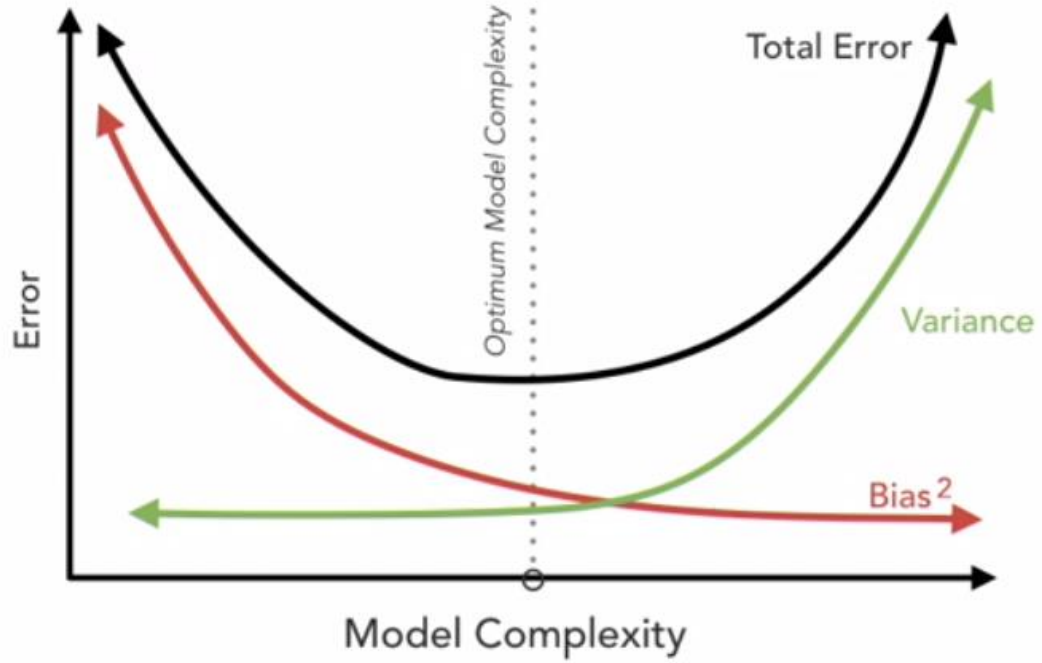
Total Error = (Bias + Variance) + Irreducible Error

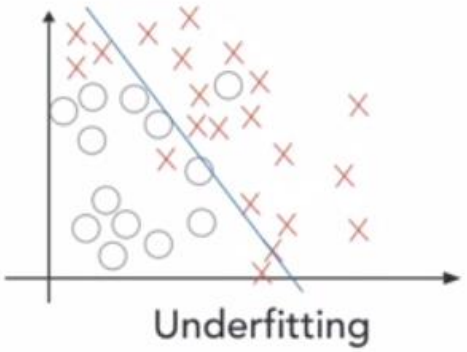


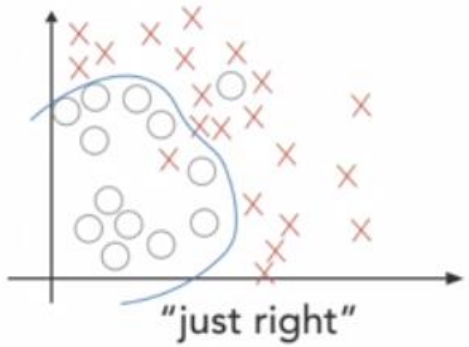
Overfitting occurs when an algorithm fits too closely to a limited set of data.

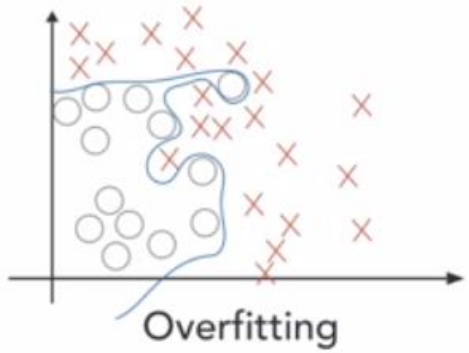
Total Error = (Bias + Variance) + Irreducible Error

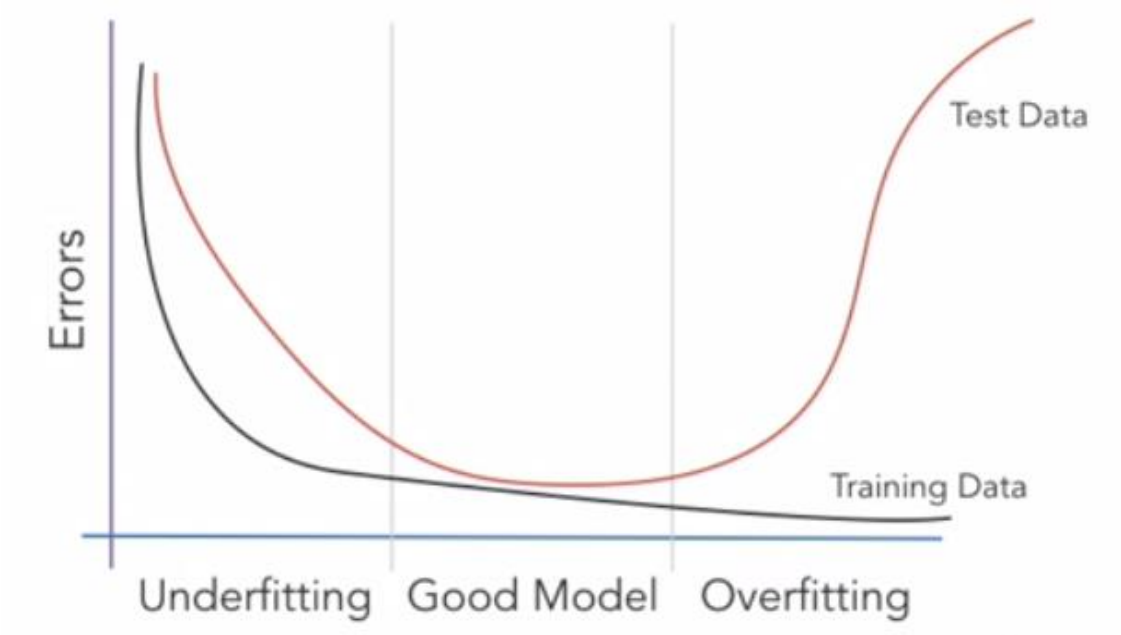



Error
Optimum Model Complexity
Total Error
Variance
Bias 2
Model Complexity

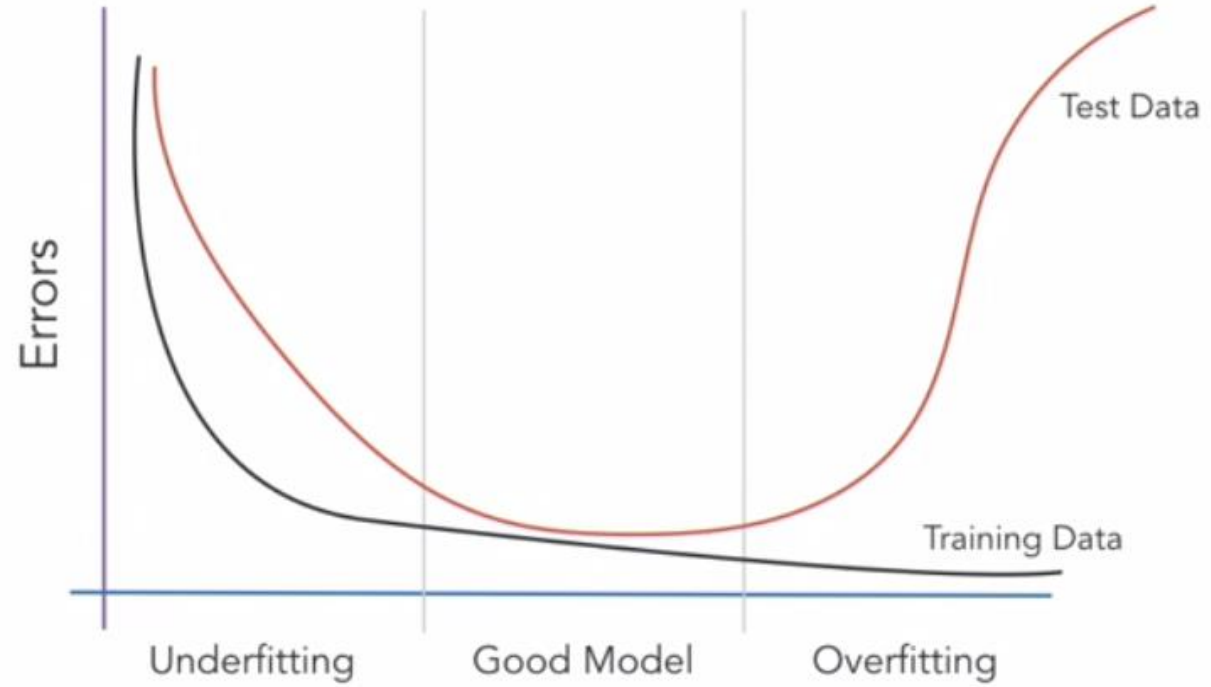

Underfitting

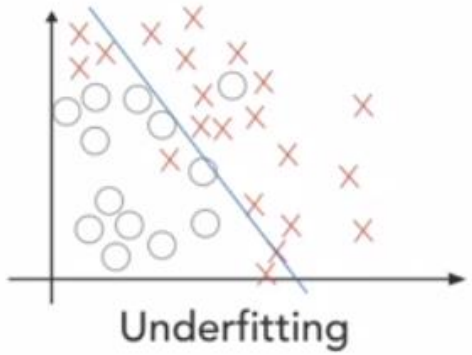

"just right"

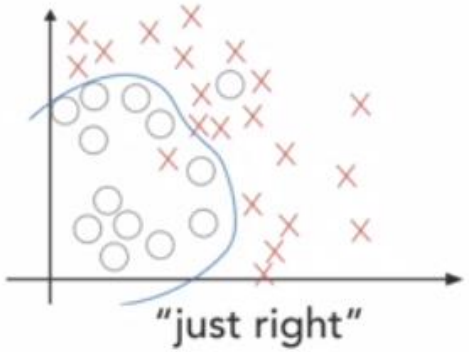

Overfitting

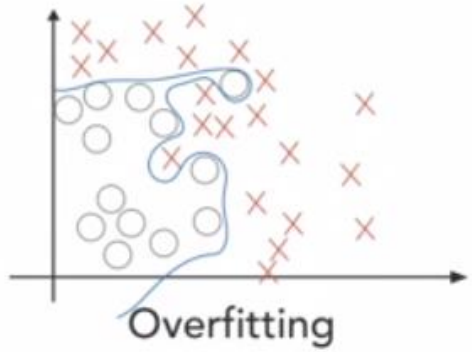


	Underfitting	Good Model	Overfitting
Complexity	Low	Medium	High
Bias	High	Low	Low
Variance	Low	Low	High
Train Error	High	Low	Low
Test Error	High	Low	High


Errors
Test Data
Training Data
Underfitting
Good Model
Overfitting


Underfitting


"just right"


Overfitting

Two Primary ways to Tune the Model for Optimal Complexity

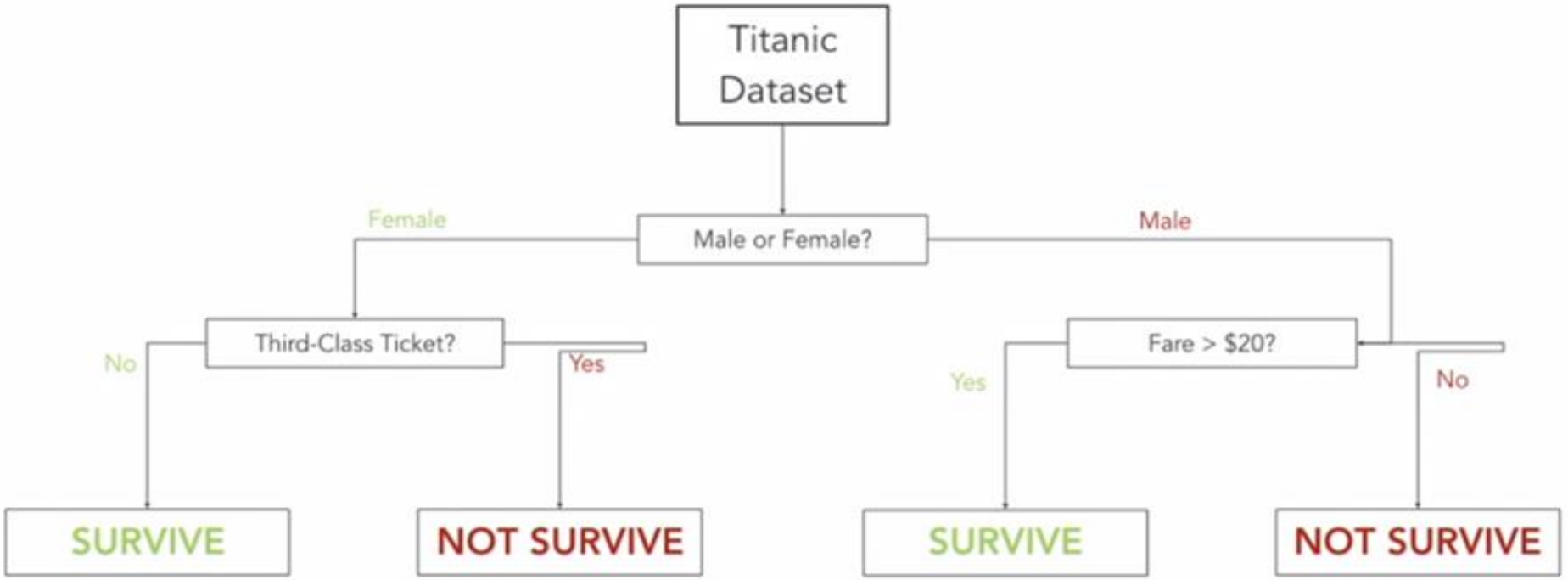
1. Hyperparameter tuning – choosing a set of optimal hyperparameters for fitting an algorithm
2. Regularization – technique used to reduce overfitting by discouraging overly complex models in some way

A model **parameter** is a configuration variable that is internal to the model and whose value can be estimated from data.

A model **hyperparameter** is a configuration that is external to the model, whose value cannot be estimated from data, and whose value guides how the algorithm learns parameter values from the data.

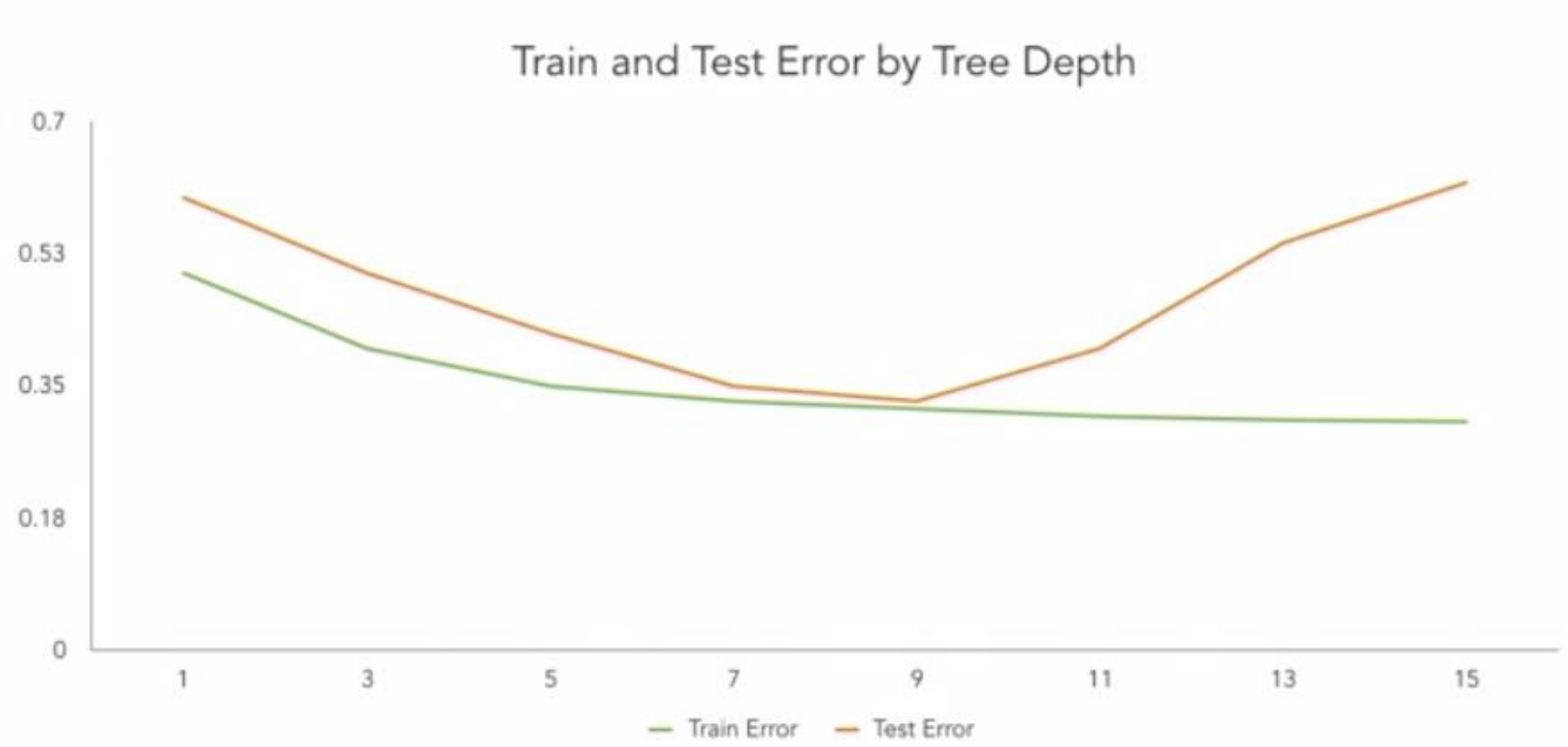
Parameters – fare = $20, ticket class = third

Hyperparameters – max depth of tree, features to consider, etc.



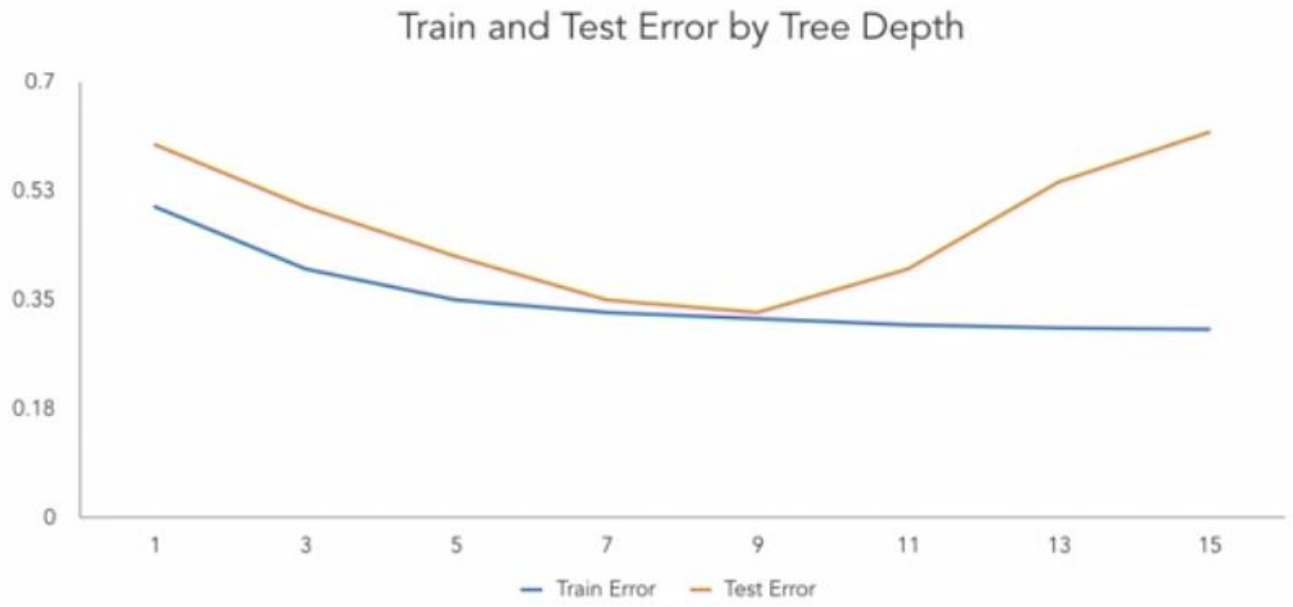
Parameters – fare = $20, ticket class = third

Hyperparameters – max depth of tree, features to consider, etc.



Two Primary ways to Tune the Model for Optimal Complexity

1. Hyperparameter tuning – choosing a set of optimal hyperparameters for fitting an algorithm



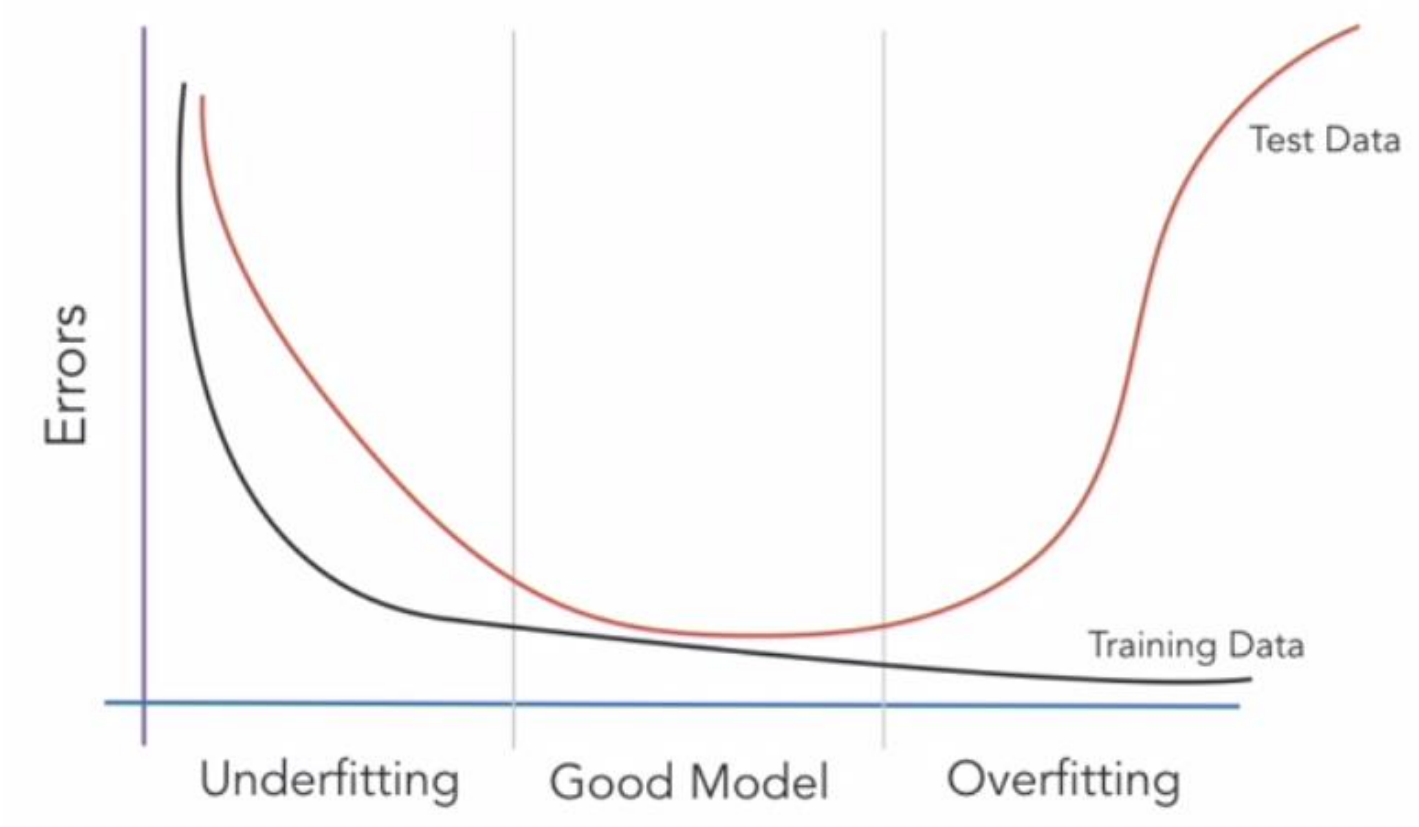
Two Primary ways to Tune the Model for Optimal Complexity

1. ~~Hyperparameter tuning – choosing a set of optimal hyperparameters for fitting an algorithm~~

2. Regularization – technique used to reduce overfitting by discouraging overly complex models in some way

Goal – allow enough flexibility for the algorithm to learn the underlying patterns in the data but provide guardrails so it doesn't overfit

Occam's Razor – Whenever possible, choose the simplest answer to a problem



Examples of Regularization

1. Ridge regression and lasso regression – adding a penalty to the loss function to constrain coefficients
2. Dropout – some nodes are ignored during training which forces the other nodes to take on more or less responsibility for the input/output