Practical Machine Learning using Apache Spark

Joy Chakraborty
Feb 25, 2017
Email: joychak1@[yahoo/gmail].com

Who am I ???

Or why does it important ???

I learn and apply ….

- Design and write software for living

- for last 17 years …

Follow at -

https://github.com/joychak/spark-prediction

Disclaimer

1. Not much of Machine Learning theory or Math
2. Not an Artificial Intelligence presentation
3. Covered Machine Learning basics for Software Engineering only
4. Introducing various content required to fulfil the job as machine learning programmer.
5. Domain specific discussion - only what demo demands

Agenda

1. Machine Learning Basics for Software Engineer?
2. Apache Spark?
3. Environment Setup
4. Writing Machine Learning application using Spark

What?

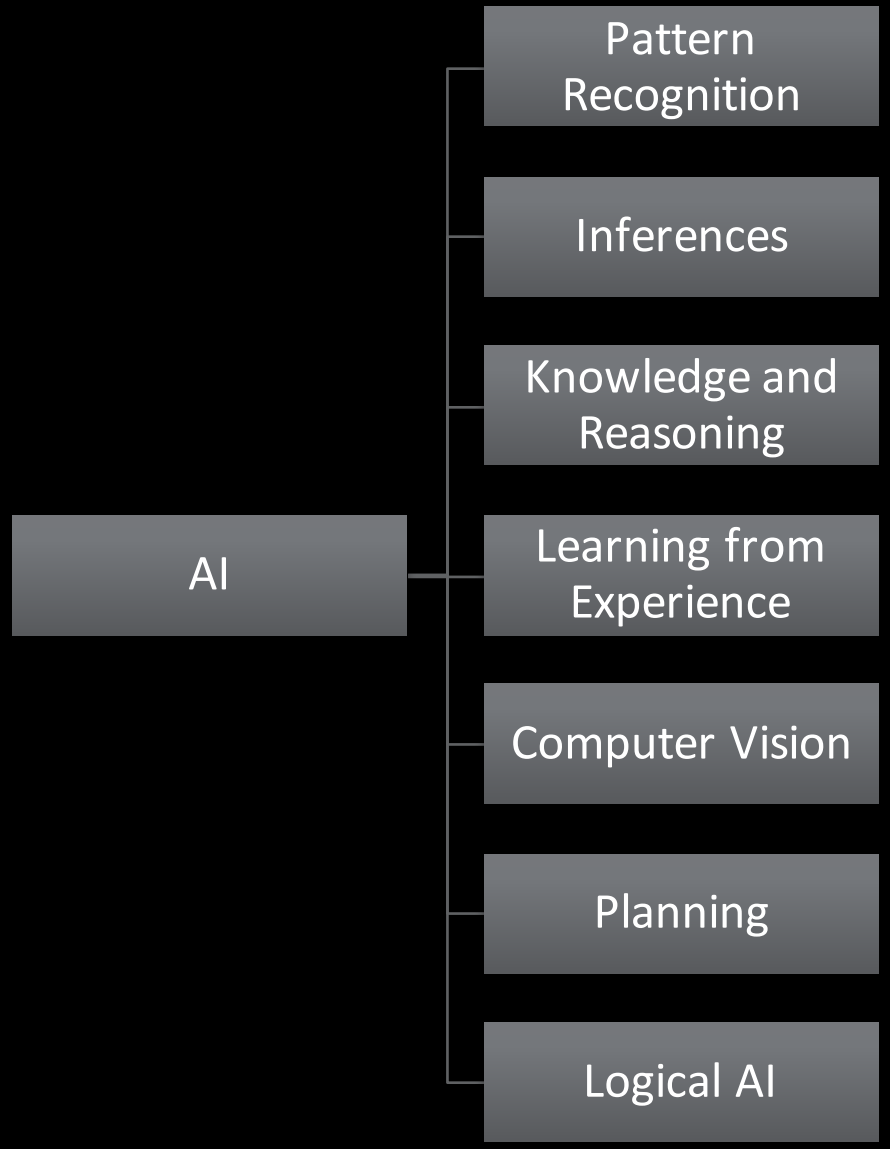
Artificial Intelligence

Branch of Computer Science - The , such as visual perception, speech recognition, decision-making, and translation between languages.

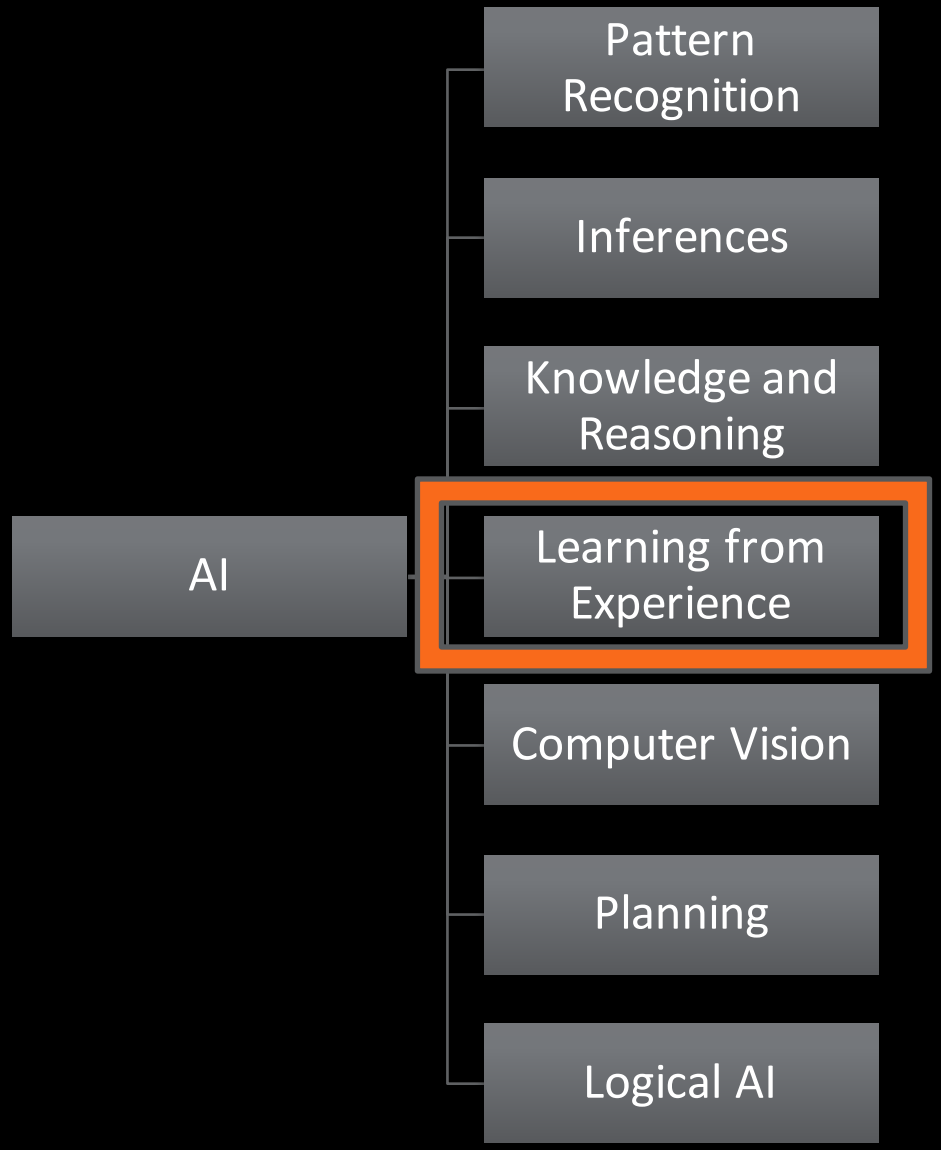
- Intelligence is the computational part of the ability to achieve goals in the world.
- AI does not have to confine itself to methods that are biologically observable

https://en.wikipedia.org/wiki/Artificial_intelligence

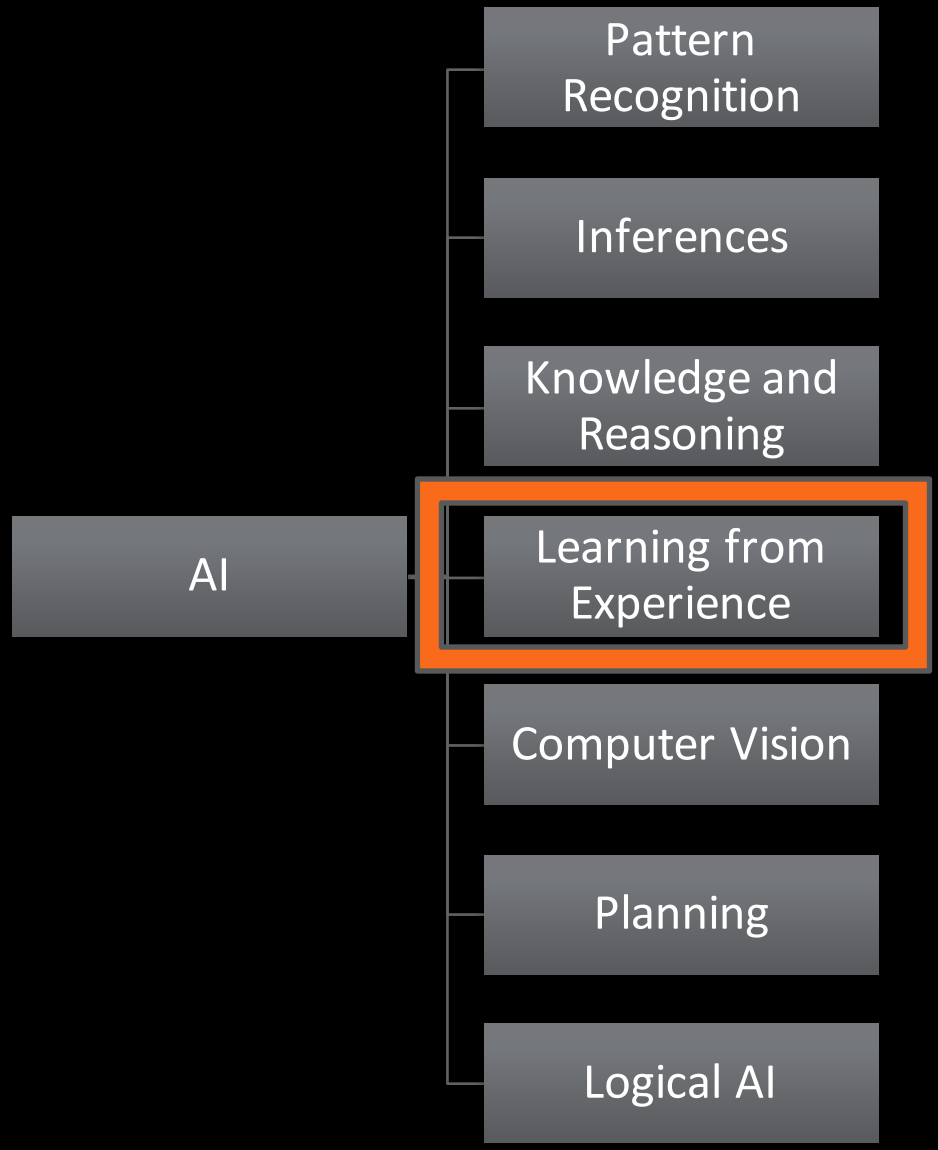
Artificial Intelligence



Artificial Intelligence



Artificial Intelligence



Machine Learning

Machine Learning

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

Machine Learning Algorithm

1. Supervised Learning
2. Unsupervised Learning

Others: Reinforcement learning, recommender systems

1. Supervised Machine Learning

Supervised learning is the **machine learning** task of inferring a function from labeled **training data**.

The training data consist of a set of training examples. In **supervised learning**, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal).

https://en.wikipedia.org/wiki/Supervised_learning

Supervised Machine Learning

Supervised Machine Learning

Supervised Machine Learning

Supervised Machine Learning

Supervised Machine Learning

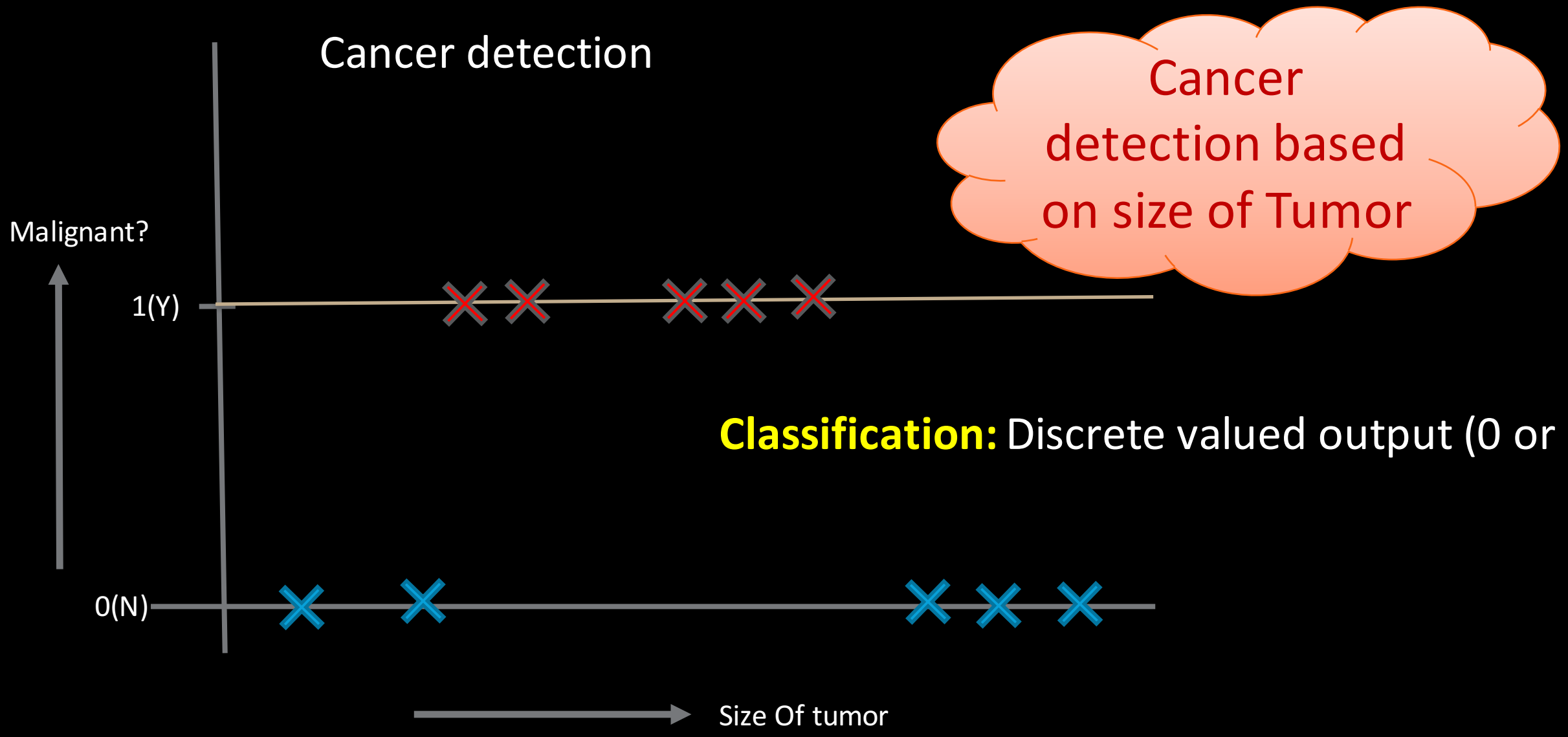
Supervised Machine Learning

Supervised Machine Learning

Supervised Machine Learning

Regression: Predict continuous valued output (price)

Supervised Machine Learning



Classification: Discrete valued output (0 or 1)

Supervised Machine Learning

Supervised Machine Learning

$$y(x) = c + \theta x$$

$$h_\theta(x) = c + \theta x$$

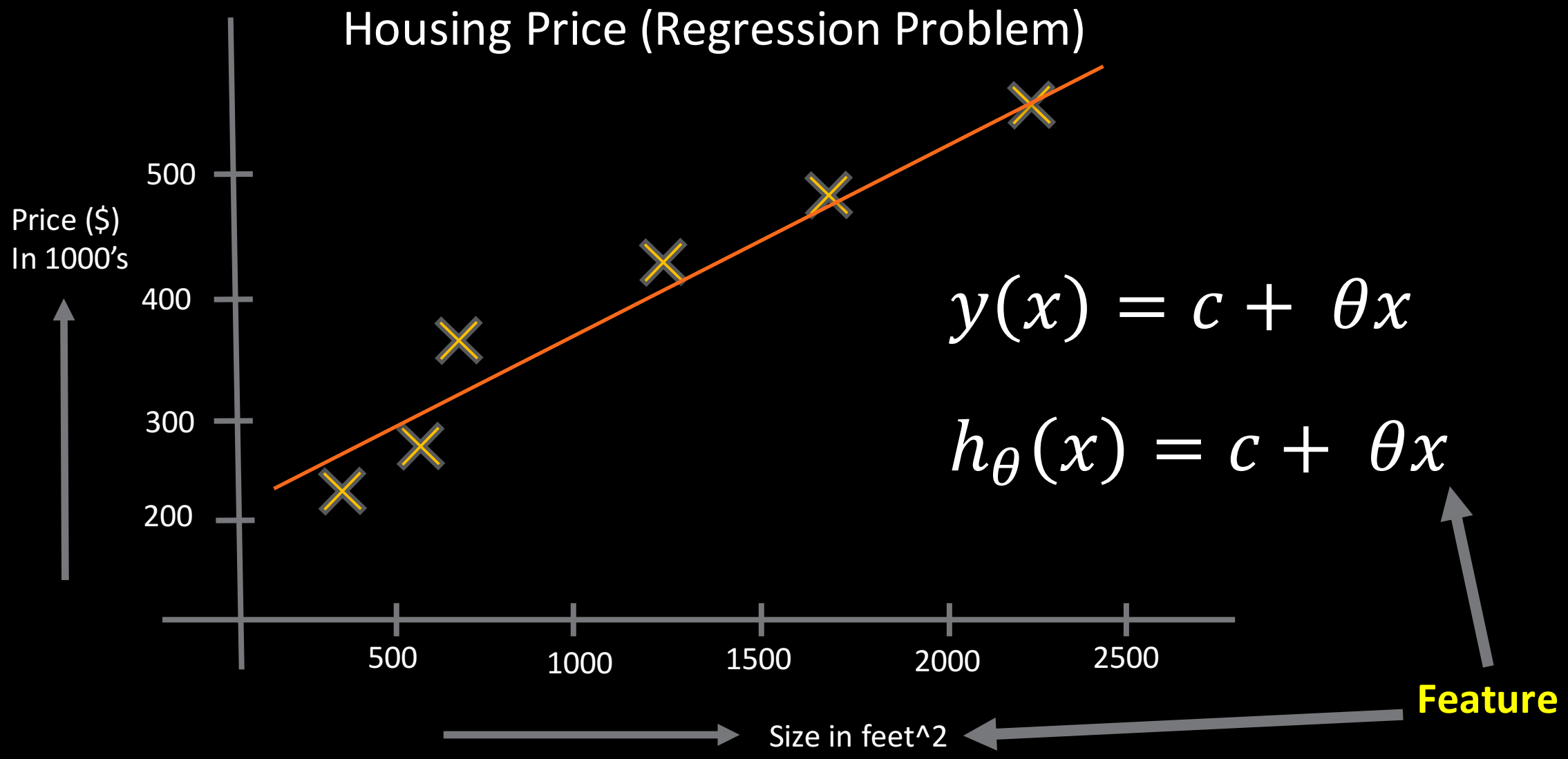
Supervised Machine Learning

$$y(x) = c + \theta x$$

$$h_\theta(x) = c + \theta x$$

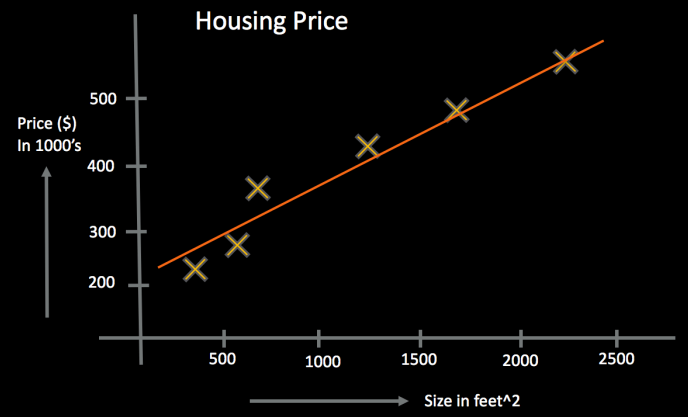
Hypothesis Function

Supervised Machine Learning



Supervised Machine Learning

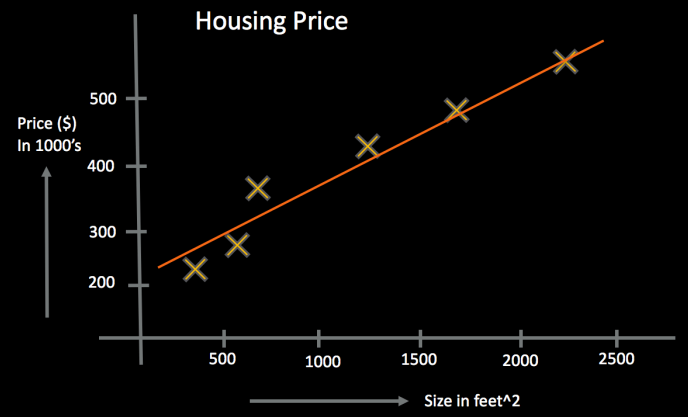
$$h_\theta(x) = c + \theta x$$



How to find the best constant and theta?

Supervised Machine Learning

$$h_\theta(x) = \theta_0 + \theta_1 x$$



How to find the best theta?

Supervised Machine Learning

$$h_\theta(x) = \theta_0 + \theta_1 x$$

If $y(x)$ is the real solution -

Supervised Machine Learning

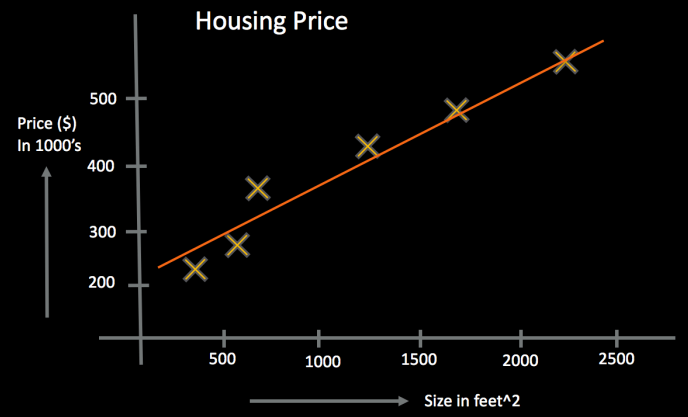
$$h_\theta(x) = \theta_0 + \theta_1 x$$

If $y(x)$ is the real solution -

$$h_\theta(x) - y(x)$$ → Objective is to Minimize

Supervised Machine Learning

$$h_\theta(x) = \theta_0 + \theta_1 x$$



$$h_\theta(x) - y(x)$$

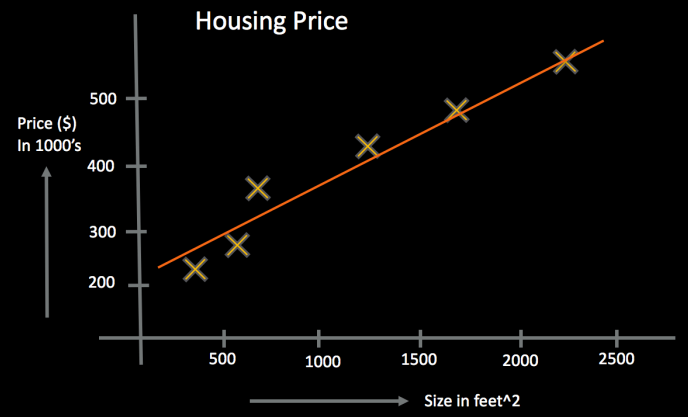
OR

$$(h_\theta(x) - y(x))^2$$

} Objective is to Minimize

Supervised Machine Learning

$$h_\theta(x) = \theta_0 + \theta_1 x$$



$$h_\theta(x) - y(x)$$

OR

$$(h_\theta(x) - y(x))^2$$

} Objective is to Minimize

$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2$$ => For all x

Supervised Machine Learning

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$h_\theta(x) - y(x)$$

$$(h_\theta(x) - y(x))^2$$

} Objective is to Minimize

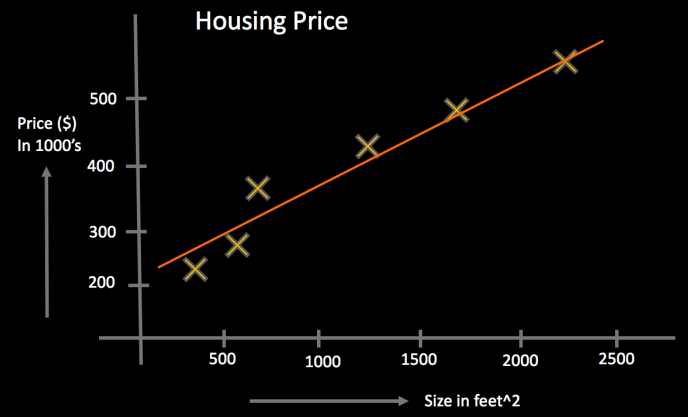
$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2 = J(\theta)$$

← Cost Function

Supervised Machine Learning

$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2 = J(\theta)$$

Objective: To minimize $J(\theta)$



Supervised Machine Learning

$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2 = J(\theta)$$

Objective: To minimize $J(\theta)$

$$\frac{d(J(\theta))}{d(\theta)} = 0$$

*** Changes in $J(\theta)$ in respect to θ

Supervised Machine Learning

$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2 = J(\theta)$$

Objective: To minimize $J(\theta)$

$$\theta_t = \theta_{t-1} - \alpha \frac{d(J(\theta))}{d(\theta)}$$

At time t At time t-1 Reduction

Start with a value of θ and then reduce by small amount and measure the difference

Supervised Machine Learning

Linearregression

$$\sum_{x=1}^{m} (h_\theta(x) - y(x))^2 = J(\theta)$$

Objective: To minimize $J(\theta)$

$$h_\theta(x) = \theta_0 + \theta_1 x$$

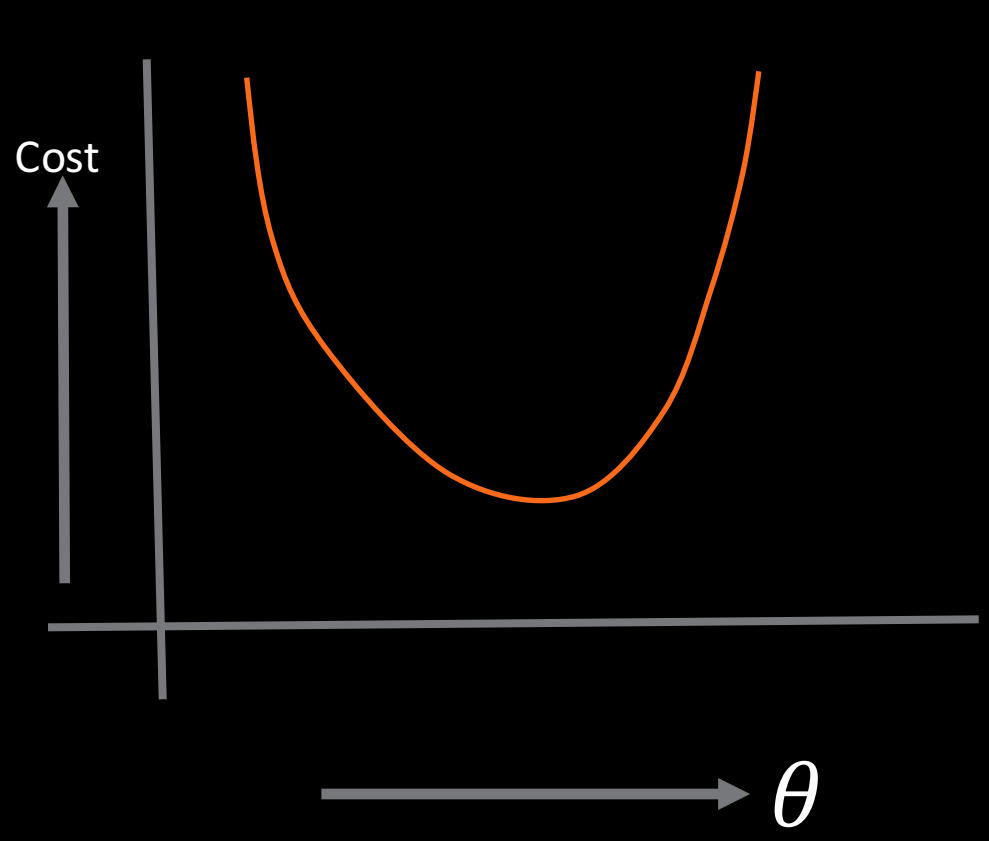
$$\theta_t = \theta_{t-1} - \alpha \frac{d(J(\theta))}{d(\theta)}$$

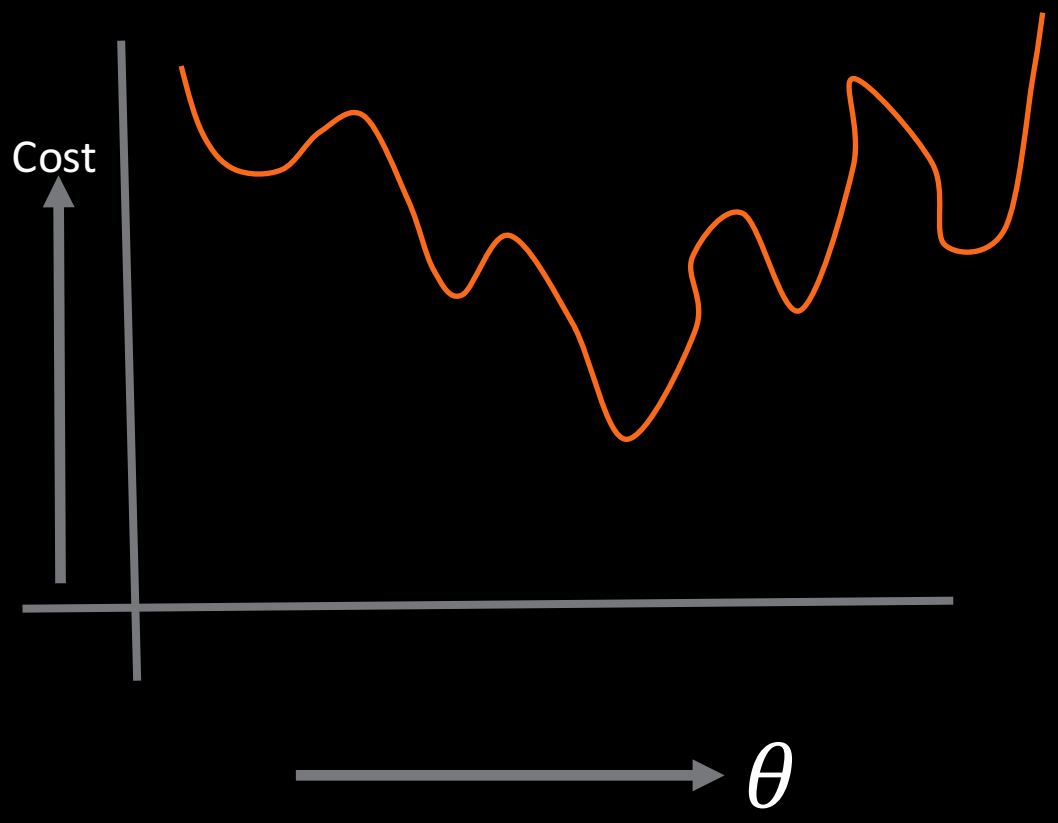
Gradient Decent (Logistic regression)

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

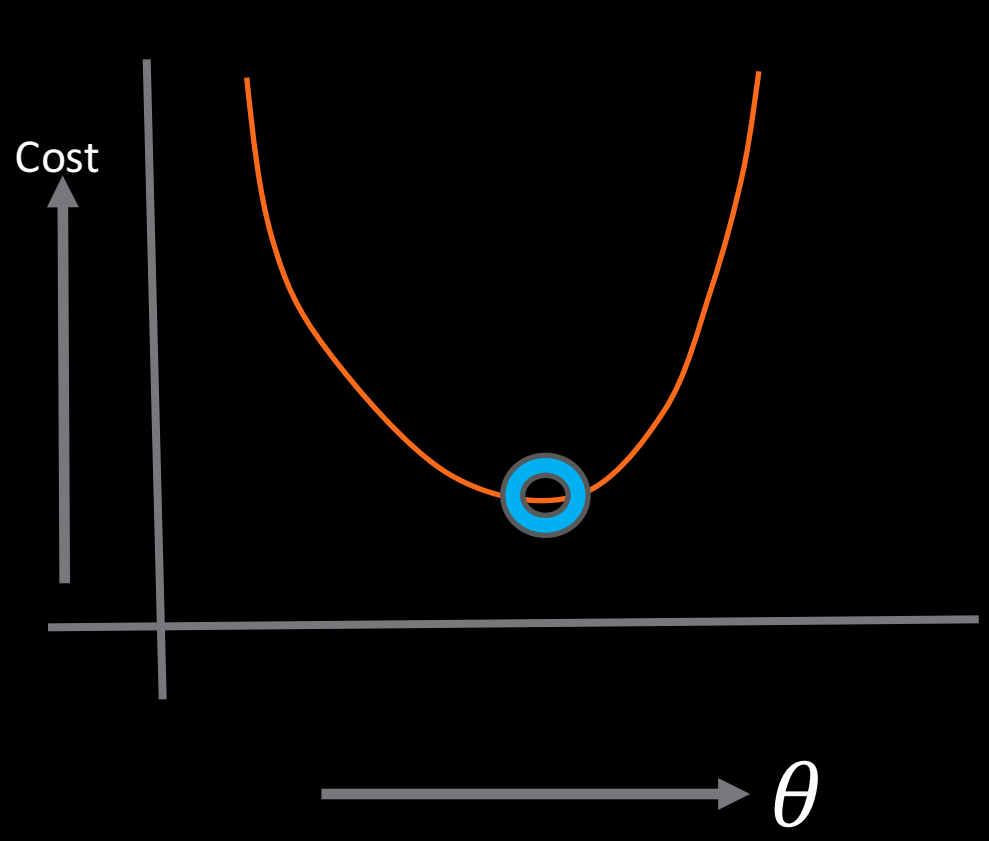
$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

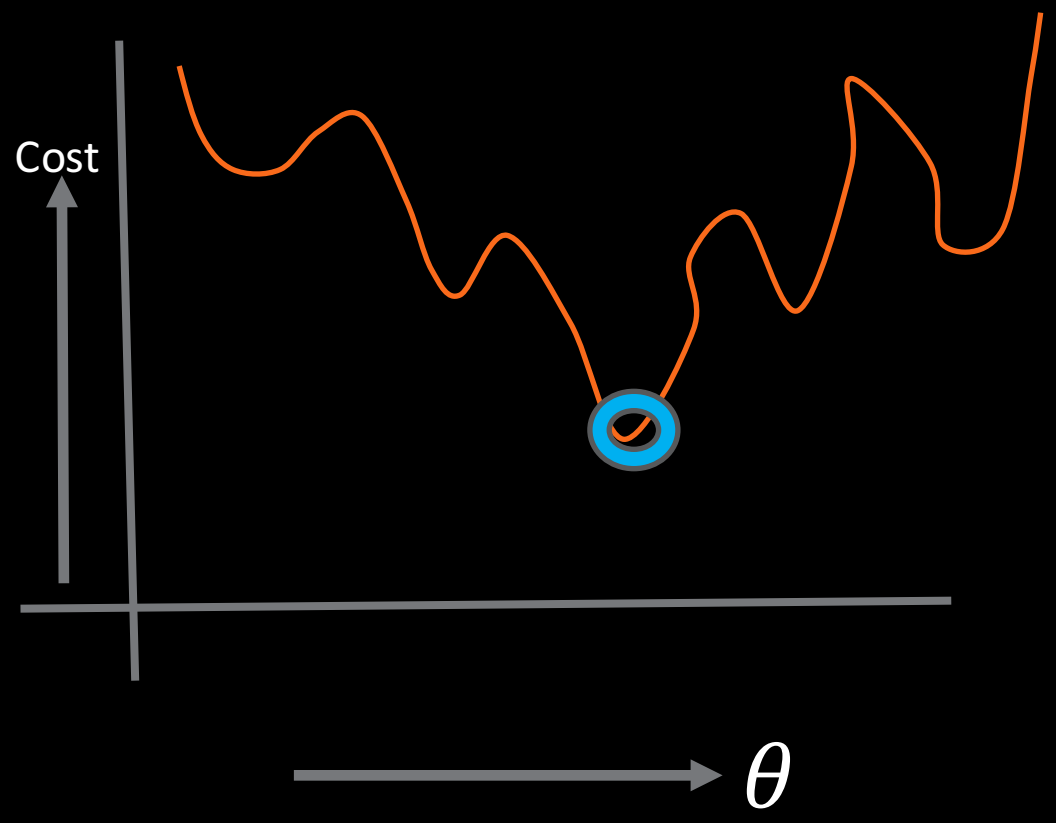
Gradient Decent



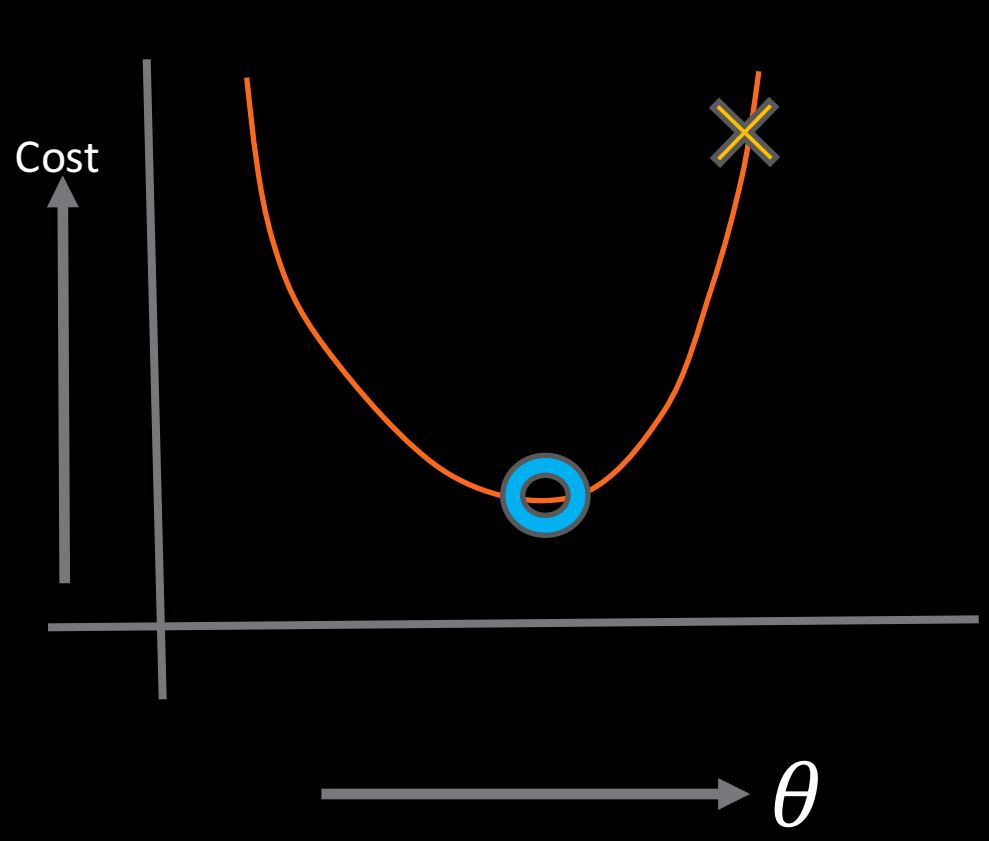


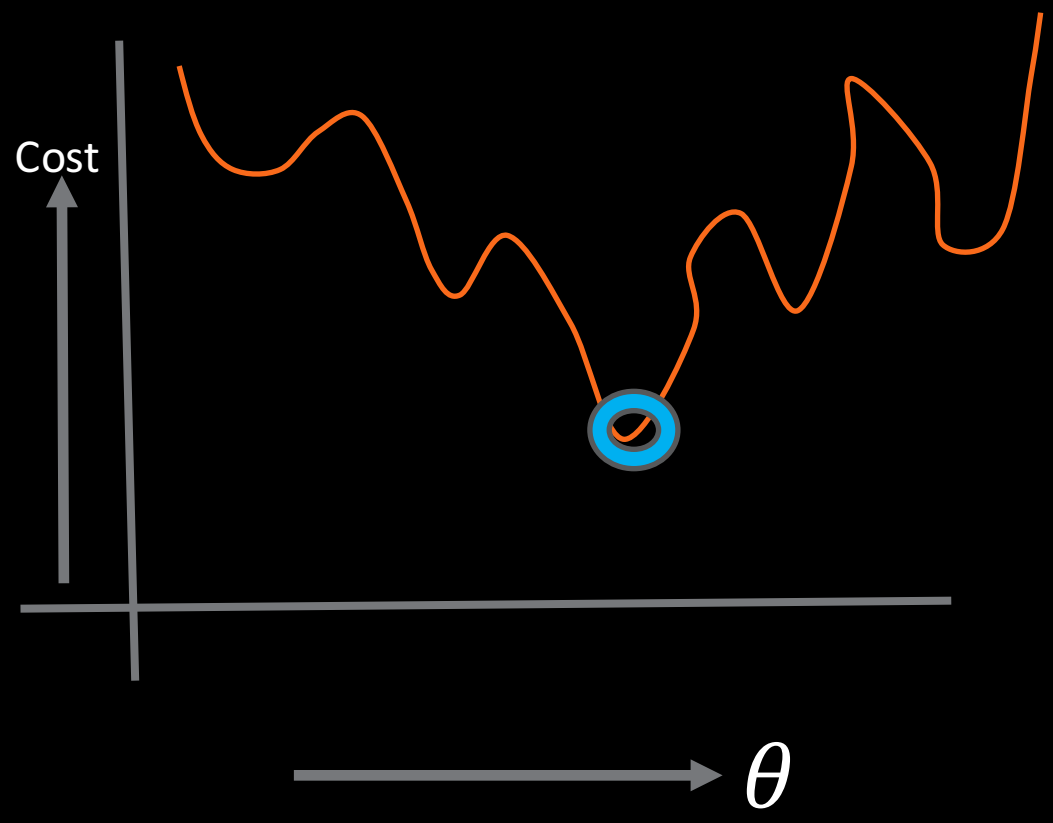
Gradient Decent



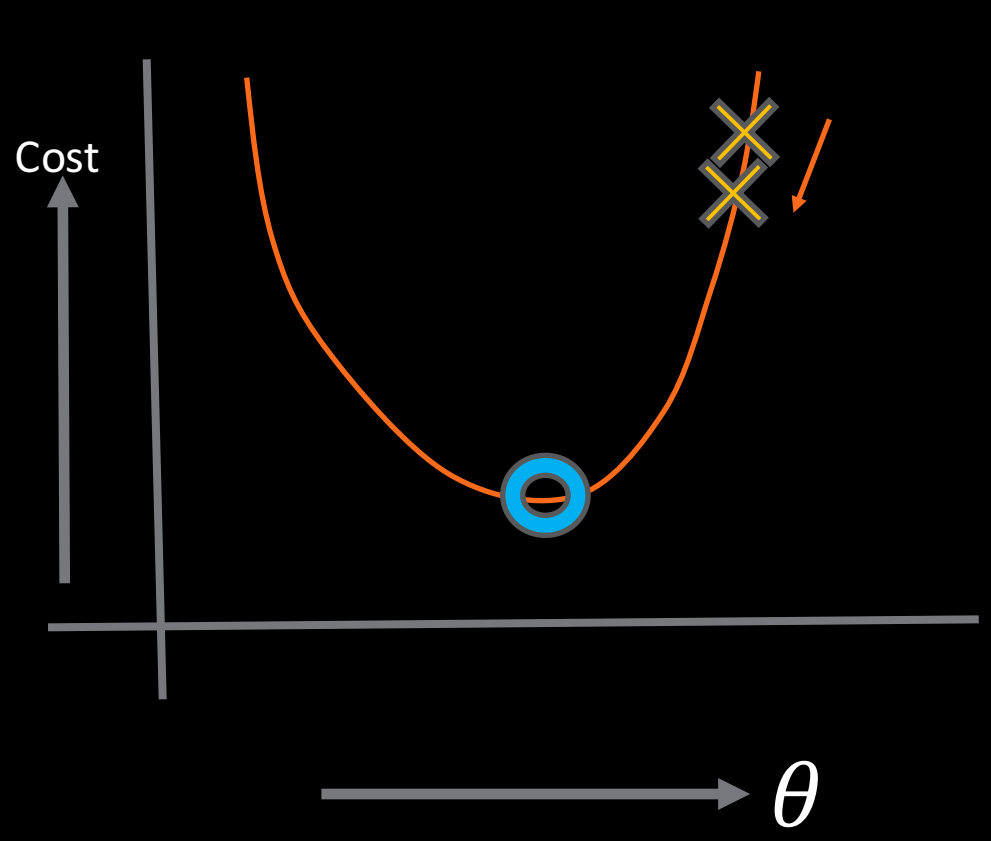


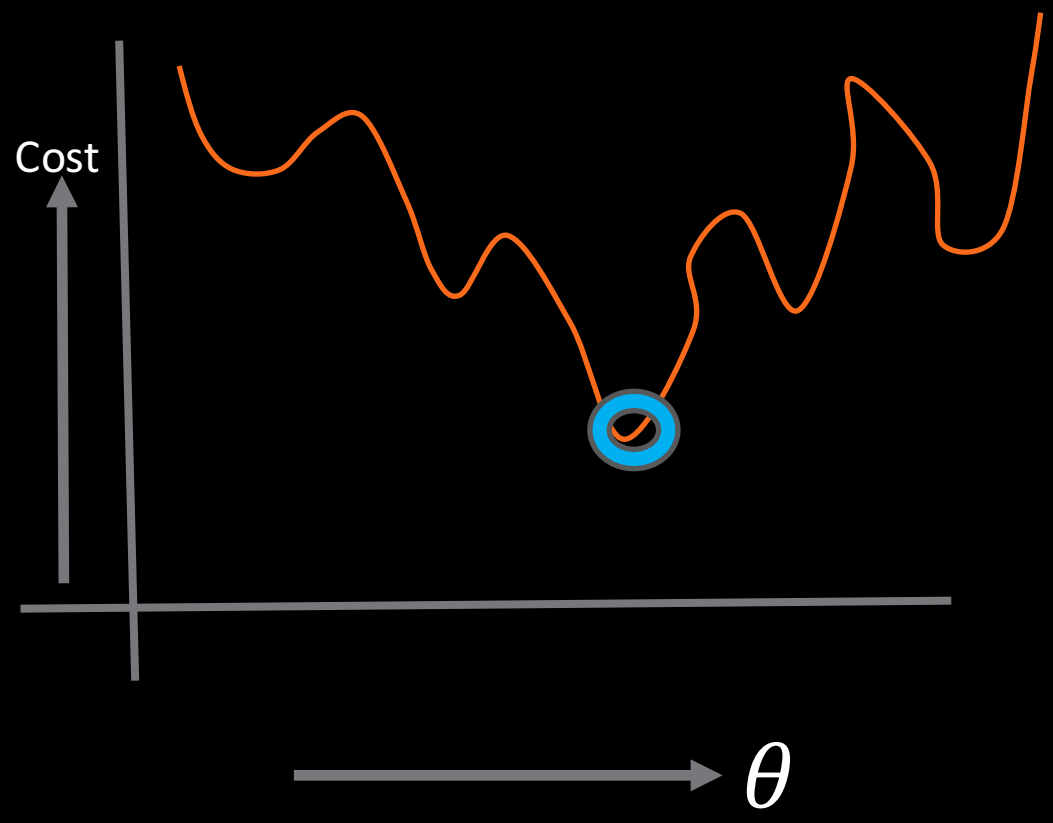
Gradient Decent



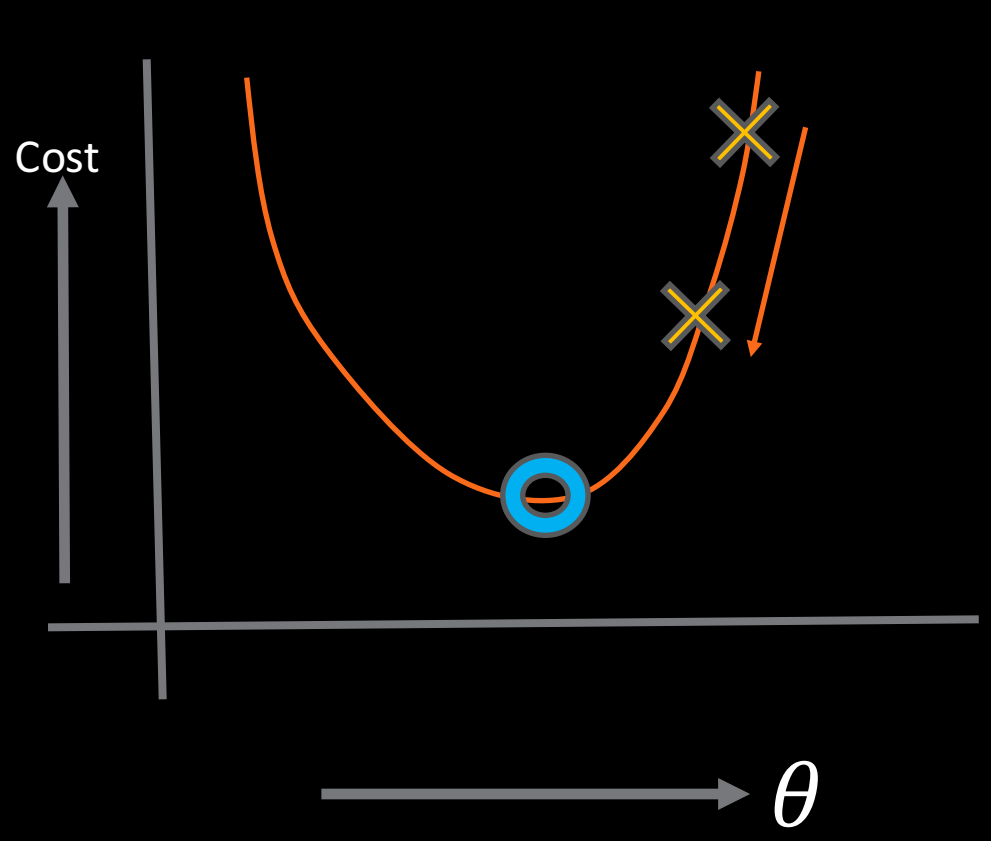


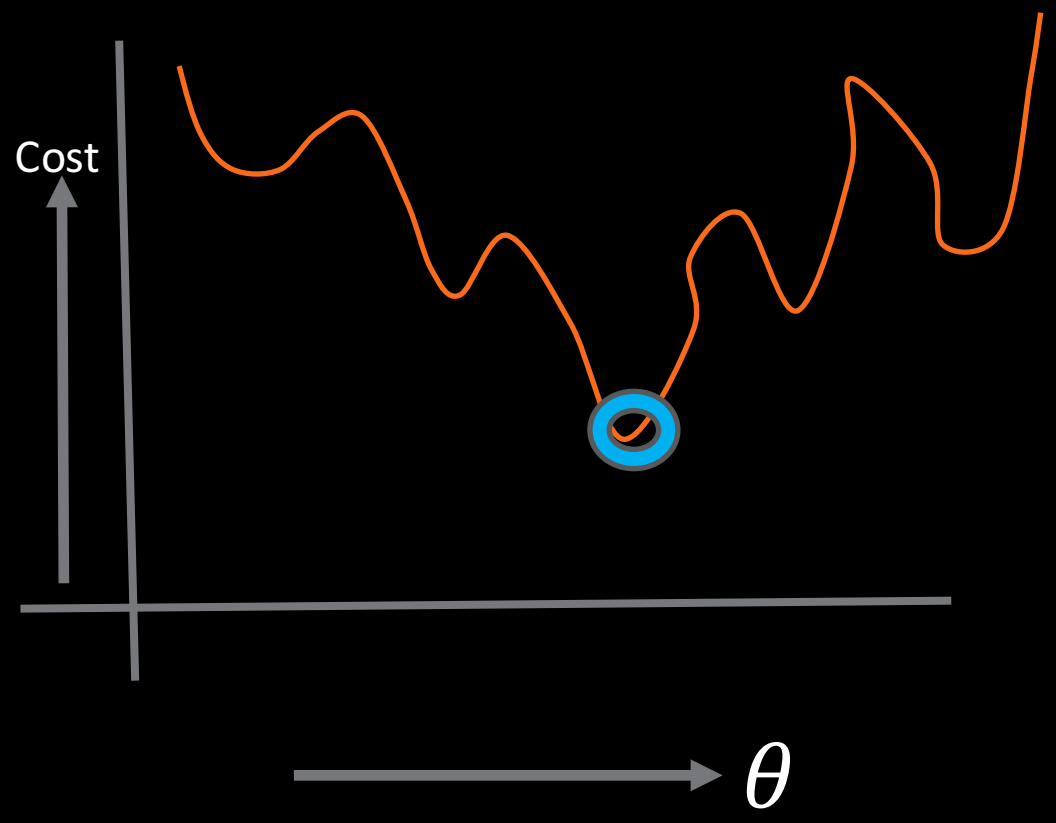
Gradient Decent





Gradient Decent





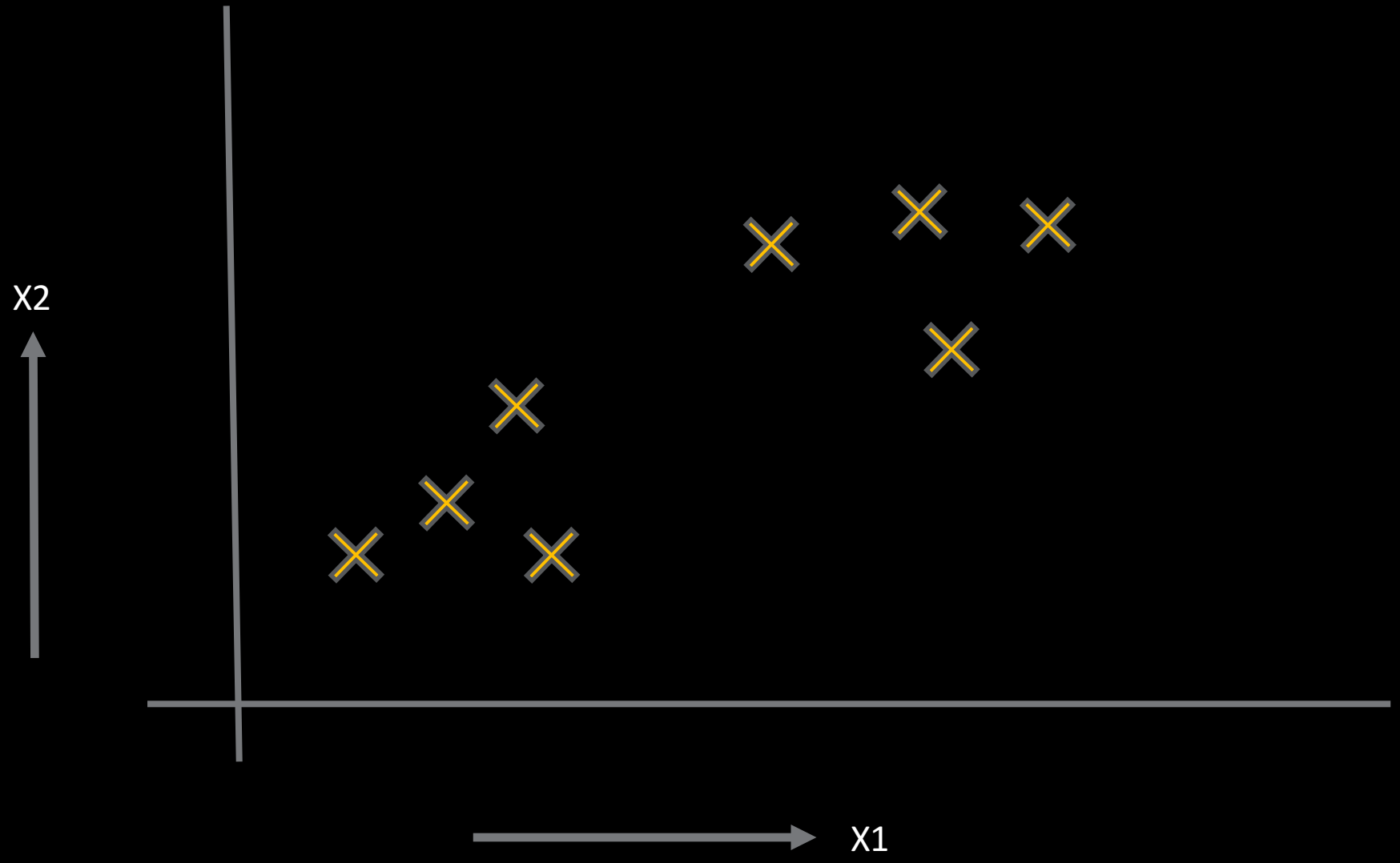
Supervised Learning

- Support Vector Machine
- Decision Tree
- Logistic Regression
- Neural Network

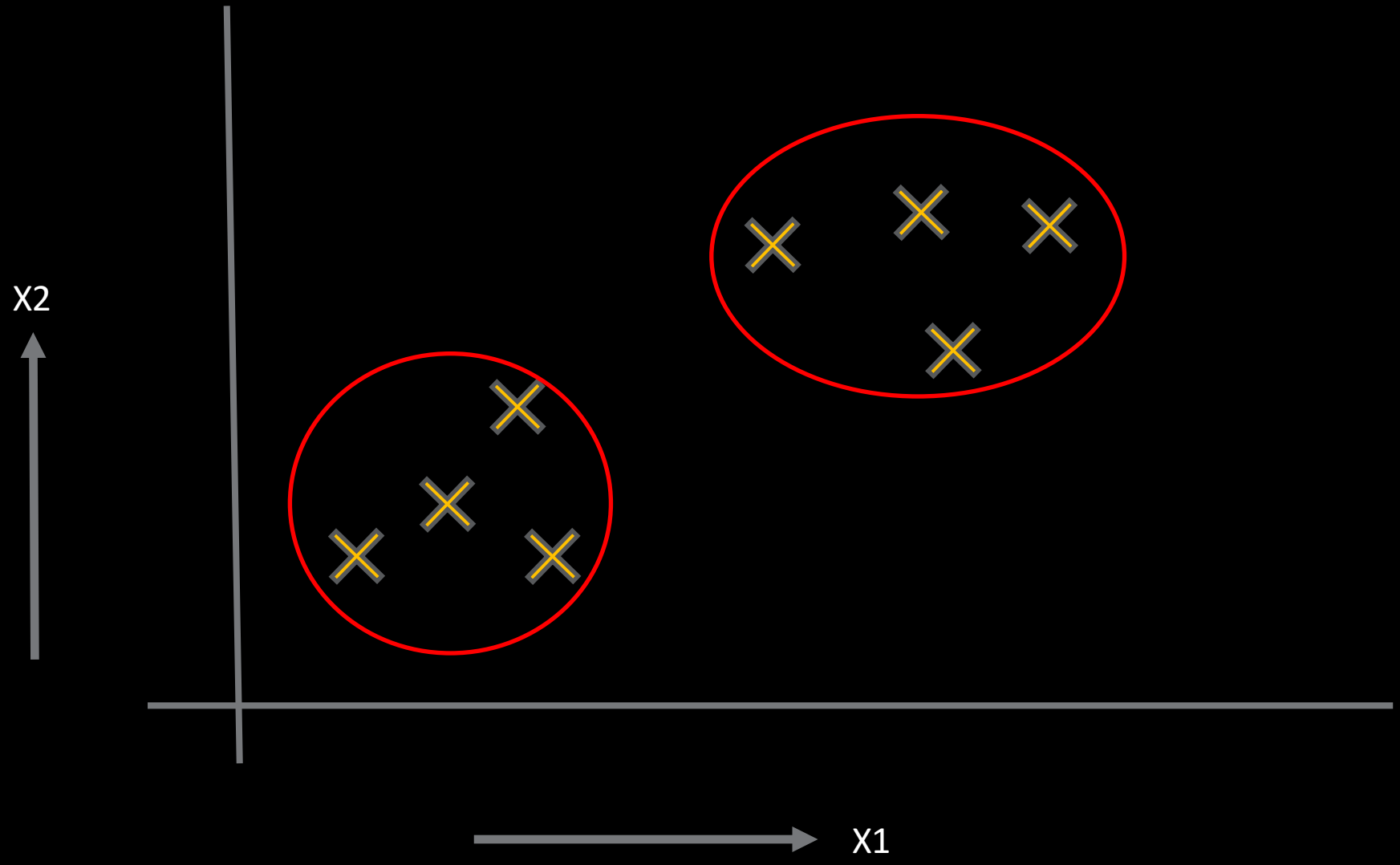
2. Unsupervised Learning

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.

Unsupervised Learning - Clustering

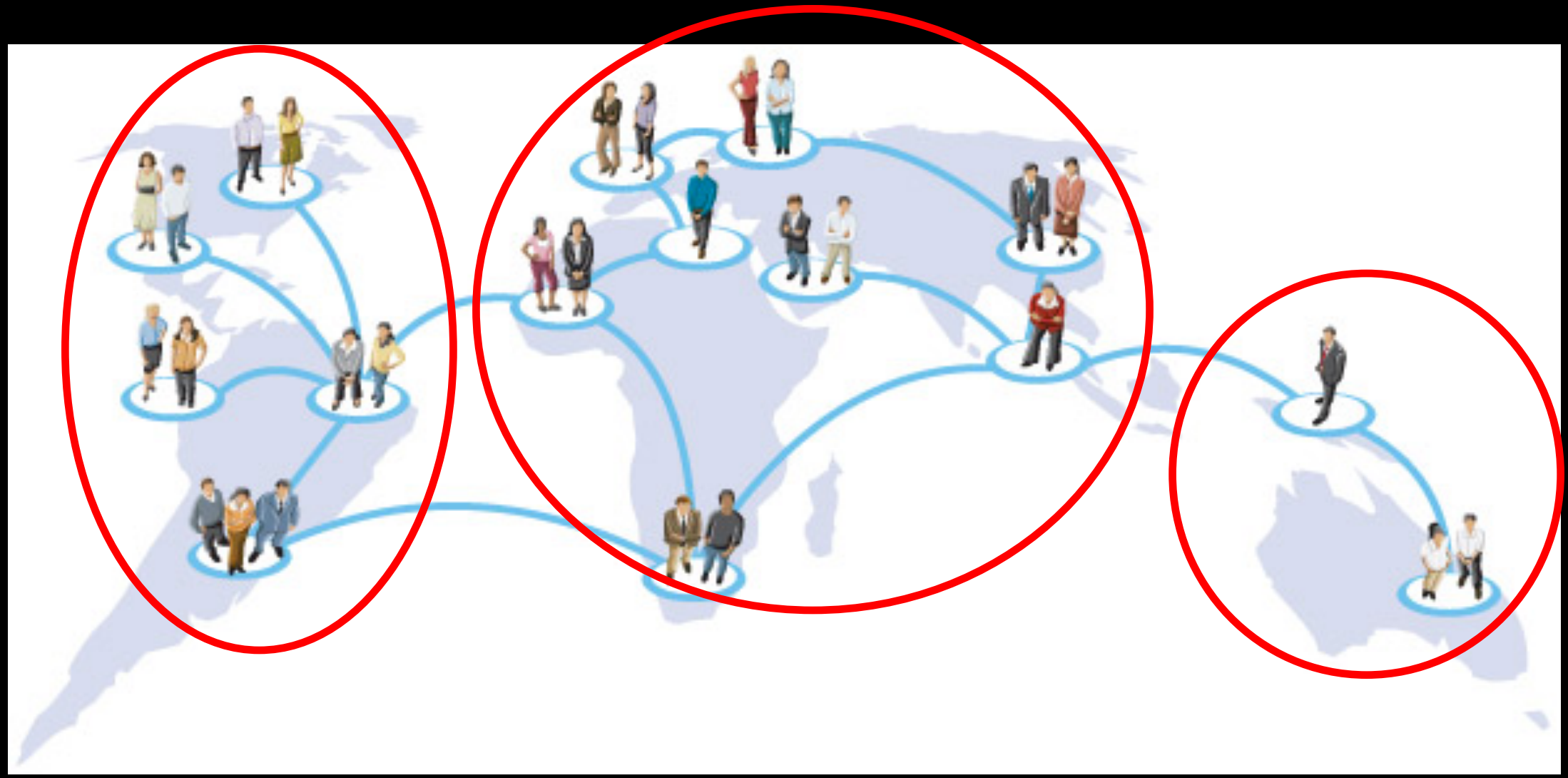


Unsupervised Learning - Clustering



Unsupervised Learning -Application

Unsupervised Learning -Application

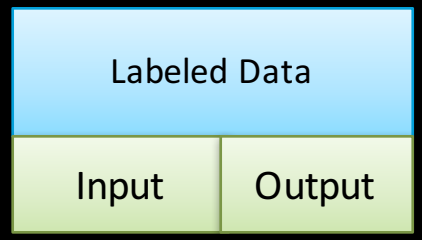


Supervised Learning

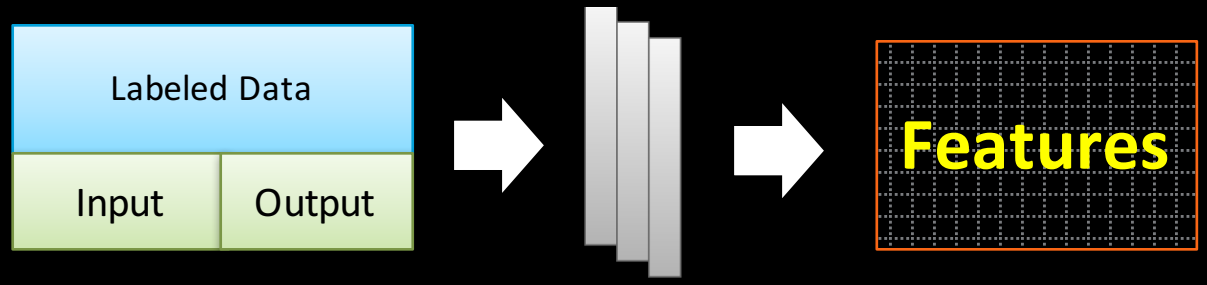
- K-means
- Non Negative Matrix Factorization (NMF)

ML - Pipeline

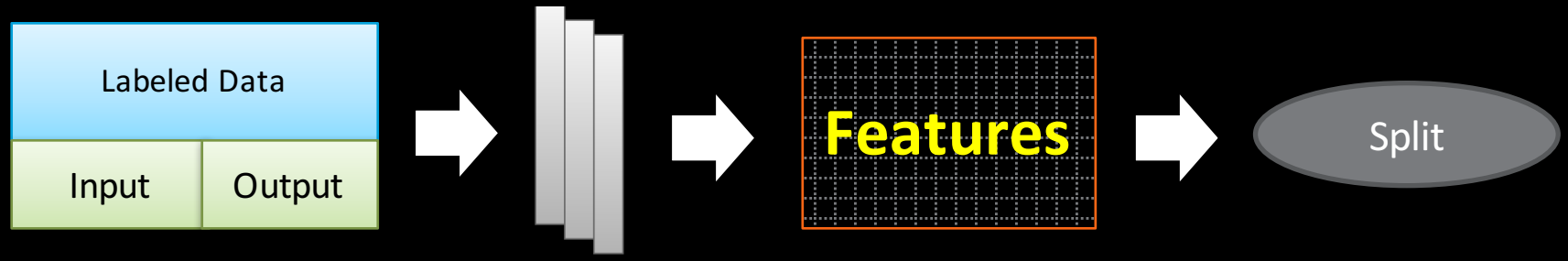
ML - Pipeline



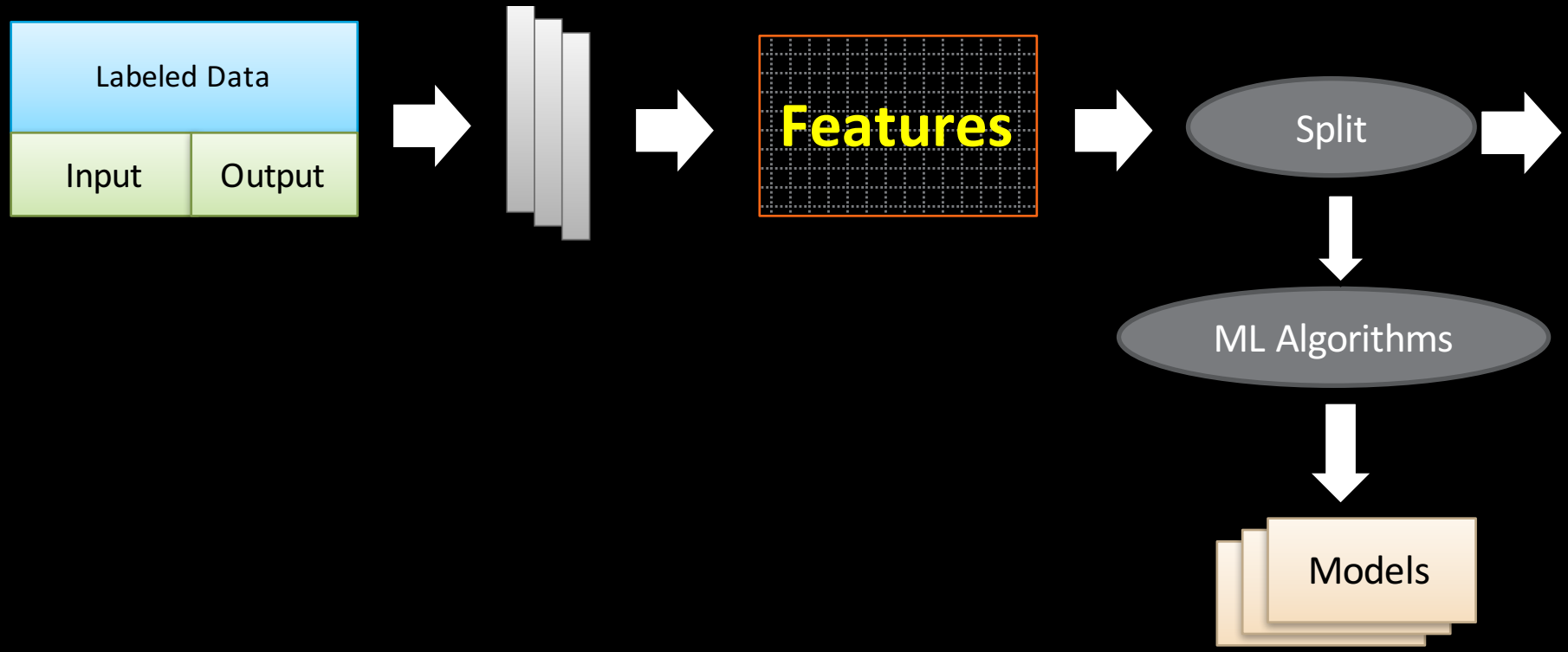
ML - Pipeline



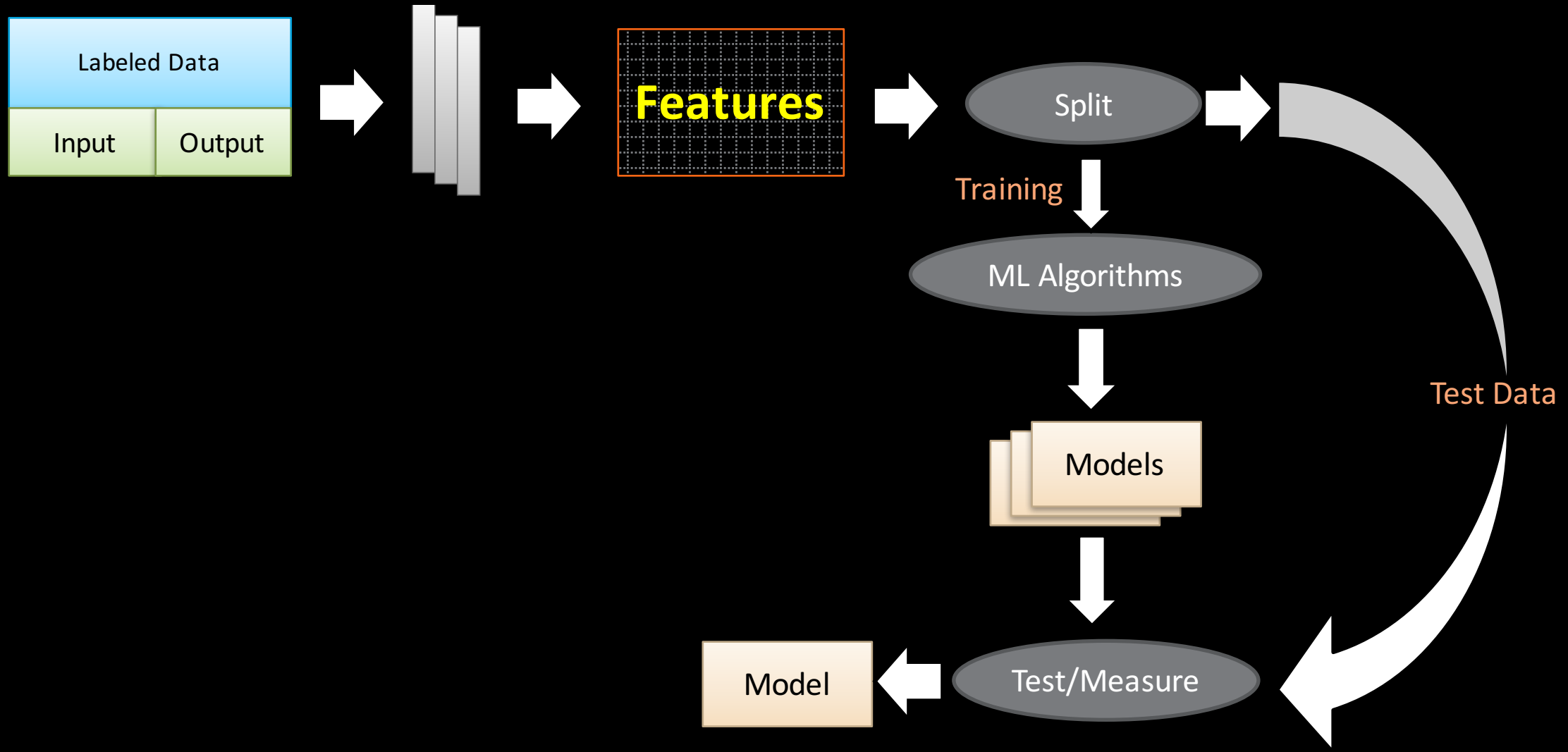
ML - Pipeline



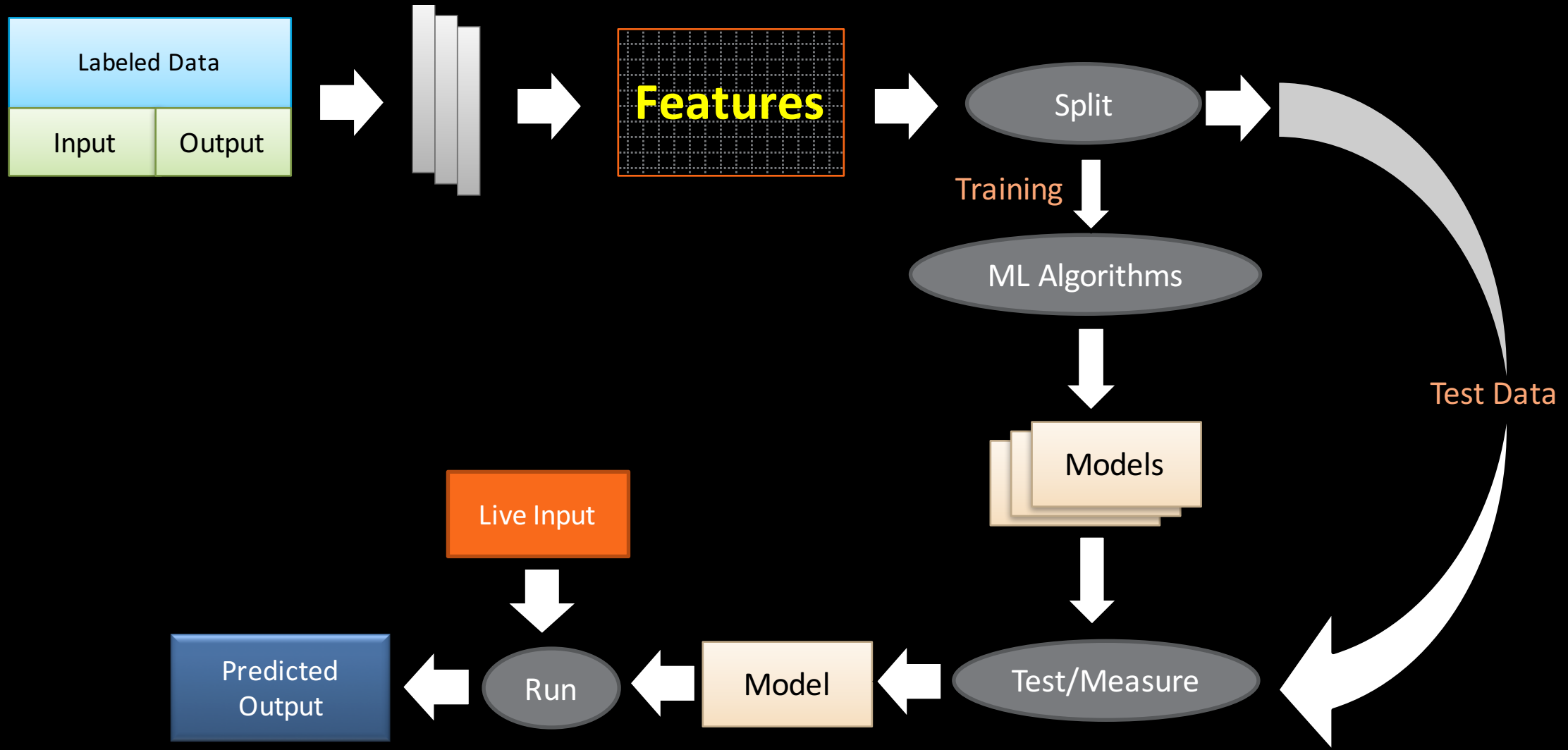
ML - Pipeline


Labeled Data
Input
Output
Features
Split
ML Algorithms
Models

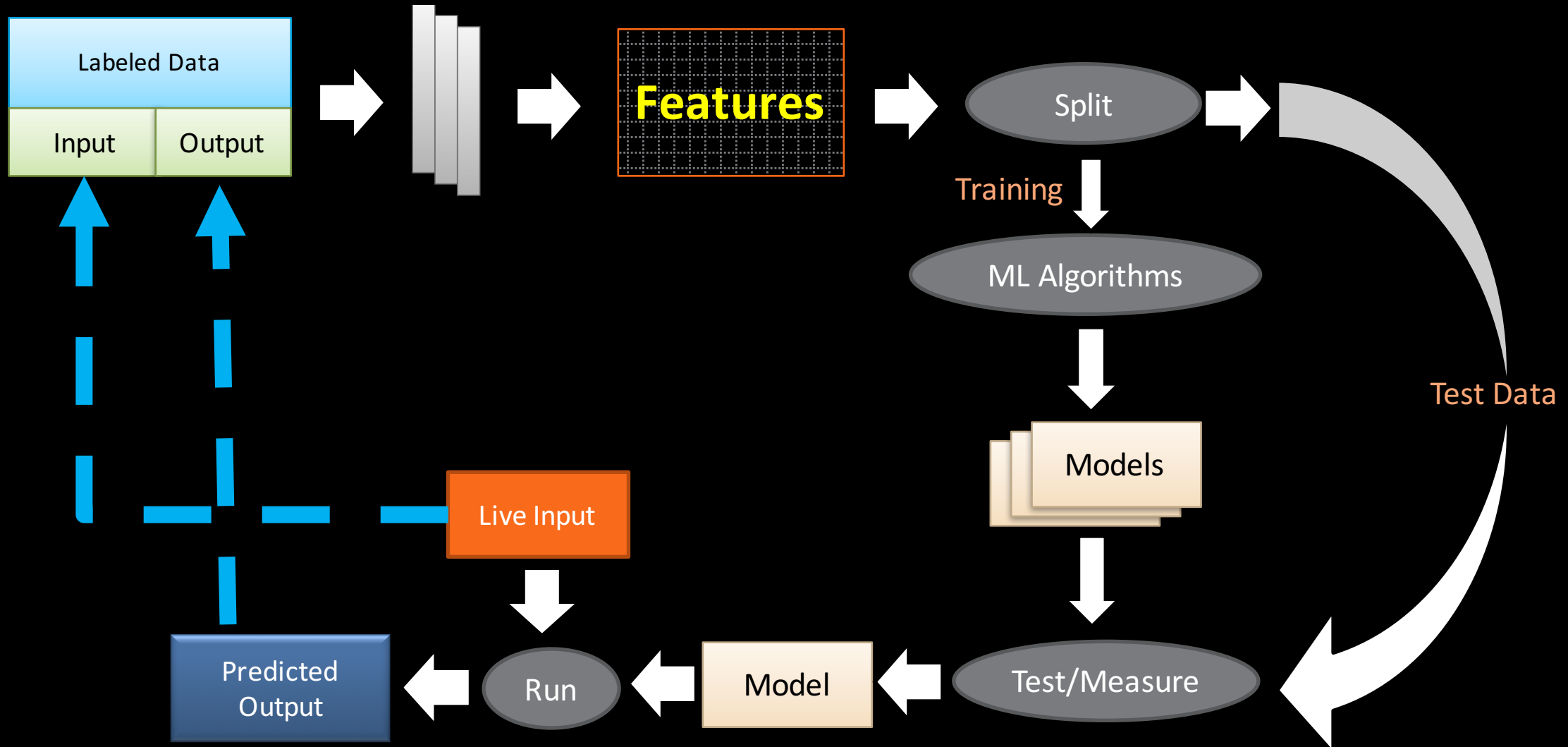
ML - Pipeline


Labeled Data
Input
Output
Features
Split
Training
ML Algorithms
Models
Test Data
Test/Measure
Model

ML - Pipeline


Labeled Data
Input
Output
Features
Split
Training
ML Algorithms
Models
Test Data
Test/Measure
Model
Run
Live Input
Predicted Output

ML - Pipeline


Labeled Data
Input
Output
Features
Split
Training
ML Algorithms
Models
Test Data
Test/Measure
Model
Run
Predicted Output
Live Input

ML – Model Performance Measurement

ML – Model Performance Measurement

		Predicted condition	
	Total population	Predicted Condition positive	Predicted Condition negative
True condition	condition positive	**True positive**	**False Negative** (Type II error)
	condition negative	**False Positive** (Type I error)	**True negative**

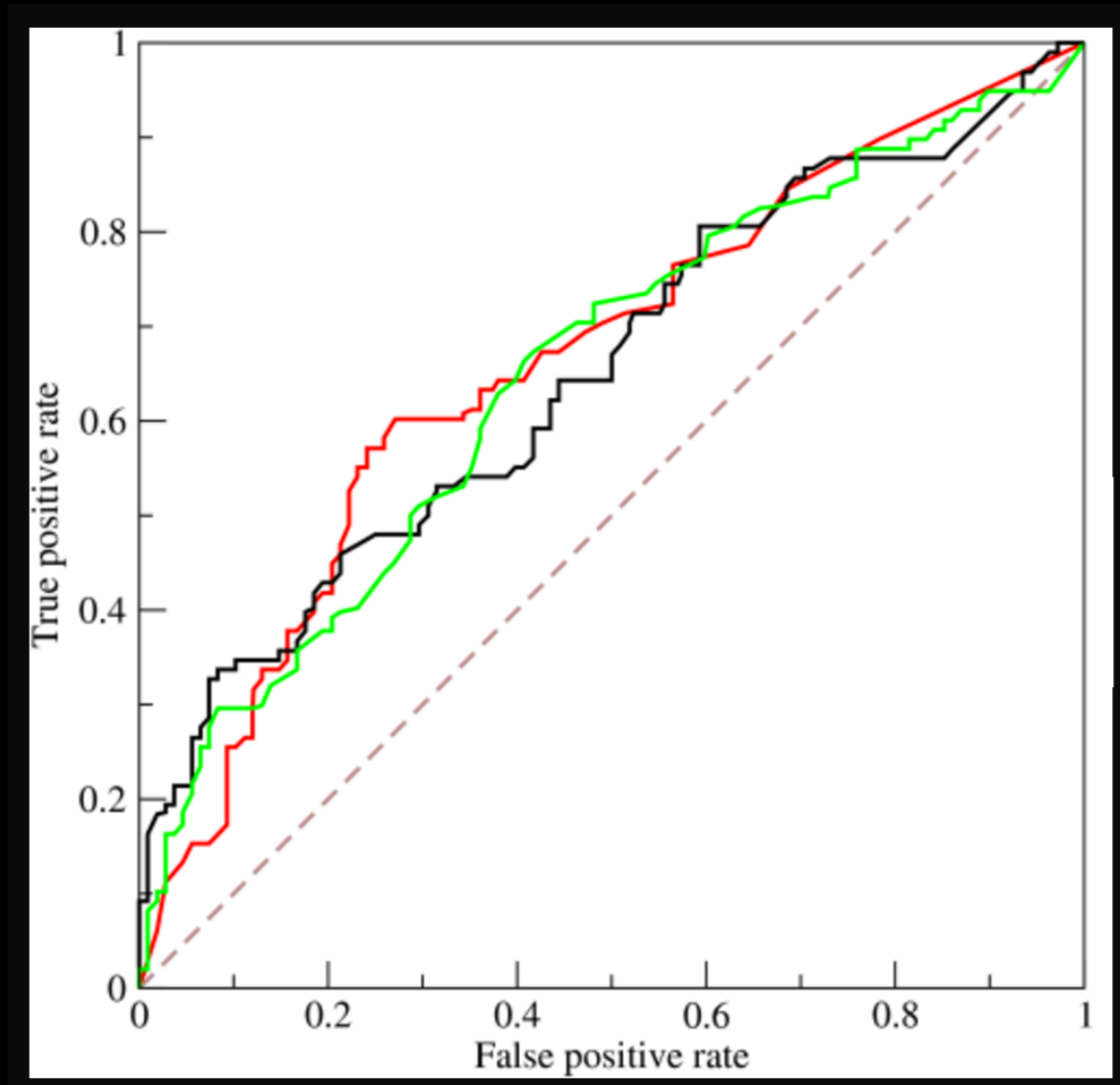
https://en.wikipedia.org/wiki/Receiver_operating_characteristic

ML – Model Performance Measurement

		Predicted condition		
	Total population	Predicted Condition positive	Predicted Condition negative	
True condition	condition positive	**True positive**	**False Negative** (Type II error)	True positive rate (TPR), Sensitivity, Recall, probability of detection $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$
	condition negative	**False Positive** (Type I error)	**True negative**	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$

https://en.wikipedia.org/wiki/Receiver_operating_characteristic

ML – Model Performance Measurement



ROC :

Receiver operating characteristic

True positive rate (TPR), Sensitivity, Recall, probability of detection

$$= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$$

False positive rate (FPR), Fall-out, probability of false alarm

$$= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$$

https://en.wikipedia.org/wiki/Receiver_operating_characteristic

How?

Big-Data

Processing Resource

Storage Resource

Big-Data– Why?

- Problems need to be solved

1. Store massive amounts of data in a cost-effective way
2. Process the data efficiently and fast

- Constraints
  - Limited storage capability per machine
  - Limited processing capability per machine

Big-Data - Hadoop

Big-Data - Hadoop

Processing Resource

Storage Resource

Map-Reduce

HDFS

Problem with Hadoop Map-Reduce

1. Lots of Disk I/O
2. Not taking advantage memory
3. In efficient networking between data nodes
4. Unnatural Programming Paradigm
5. Mostly good for batch processing not much real-time

Distributed Computing Framework – Apache Spark

Introducing Apache Spark

What is Apache Spark

Fast and expressive cluster computing system interoperable with Apache Hadoop

Improves efficiency through:

- » In-memory computing primitives
- » General computation graphs

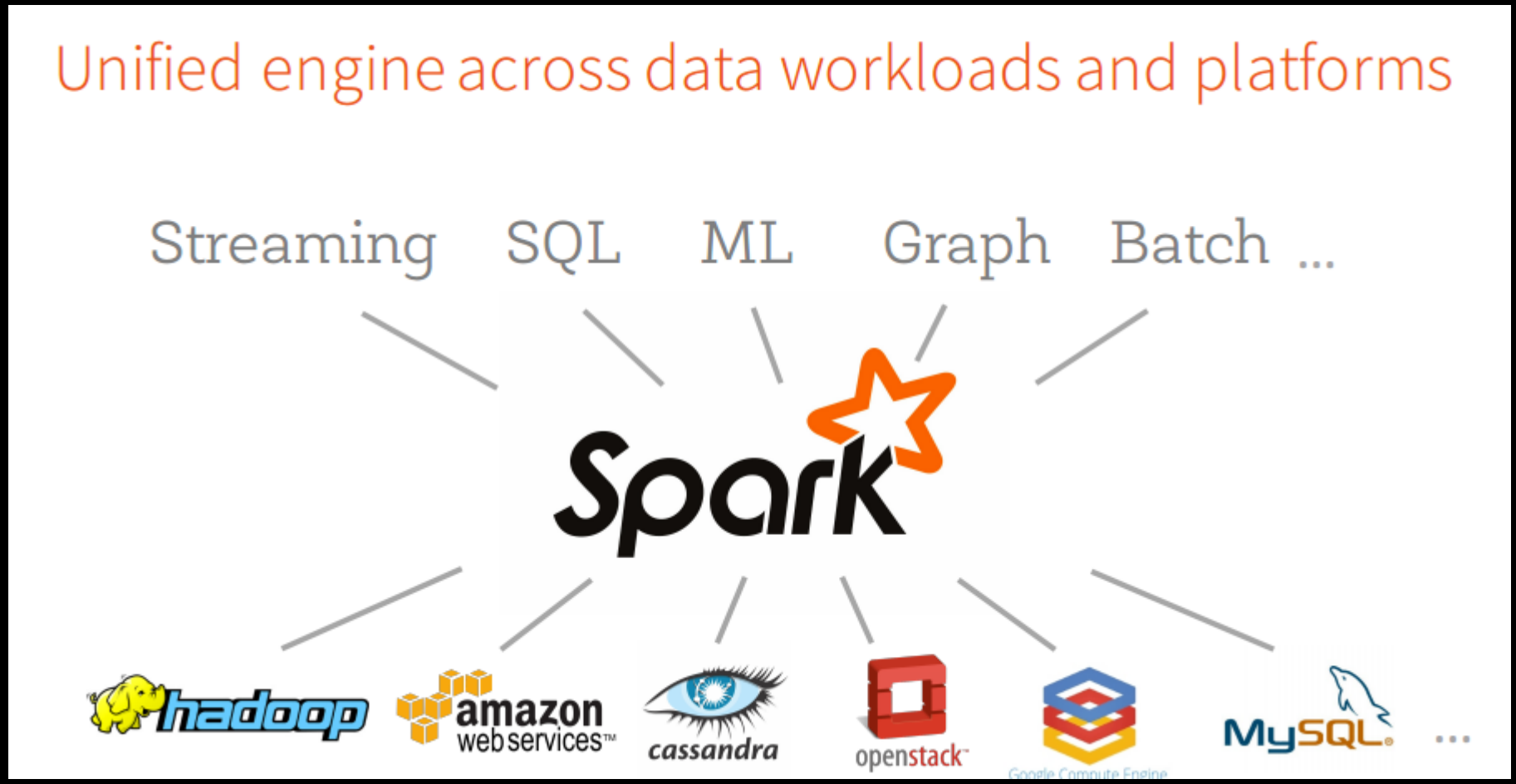
→ Up to 100× faster (2-10× on disk)

Improves usability through:

- » Rich APIs in Scala, Java, Python
- » Interactive shell

→ Often 5× less code

Apache Spark



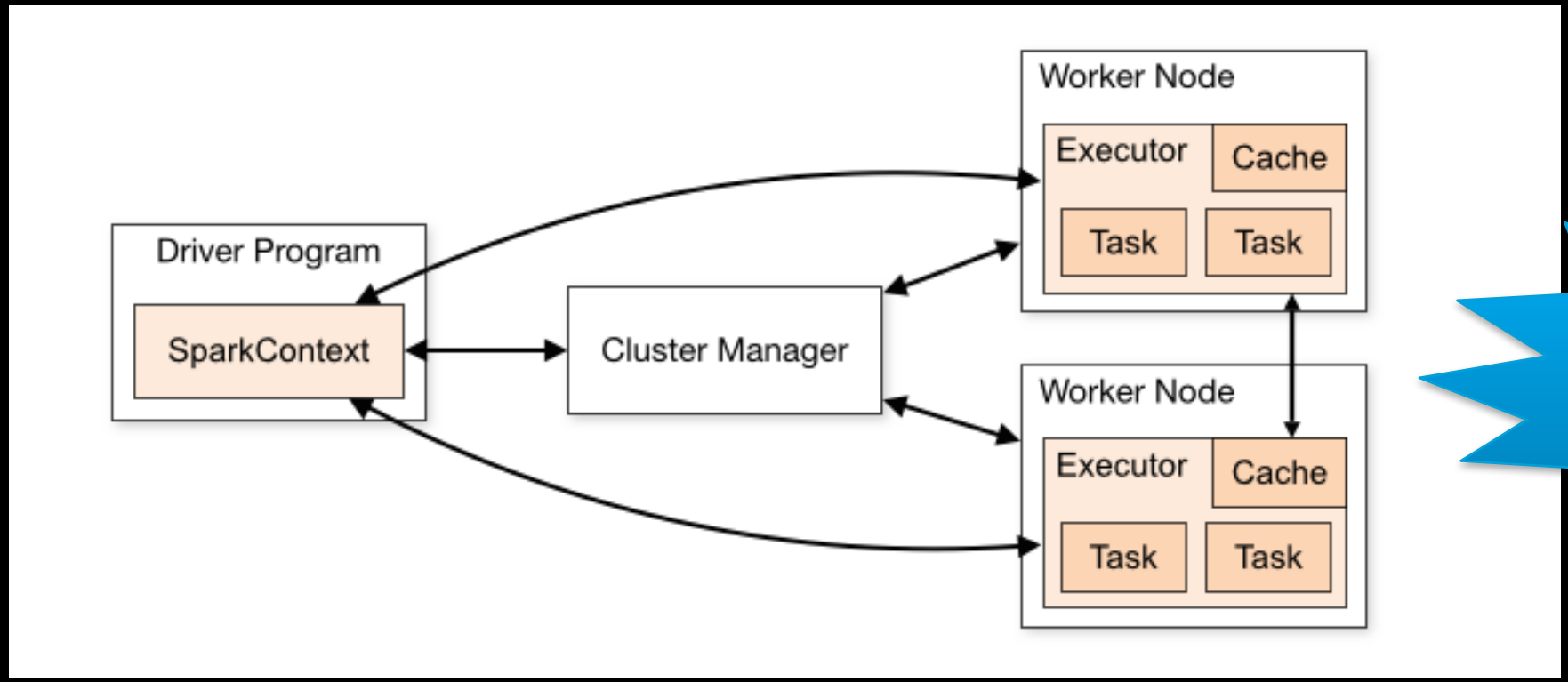
Apache Spark

Apache Spark

In case you missed our earlier blog post, using Spark on 206 EC2 machines, we sorted 100 TB of data on disk in 23 minutes. In comparison, the previous world record set by Hadoop MapReduce used 2100 machines and took 72 minutes. This means that Apache Spark sorted the same data **3X faster** using **10X fewer machines**. All the sorting took place on disk (HDFS), without using Spark's in-memory cache. This entry tied with a UCSD research team building high performance systems and we jointly set a new world record.

	Hadoop MR Record	Spark Record	Spark 1 PB
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400 physical	6592 virtualized	6080 virtualized
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	570 GB/s
Sort Benchmark Daytona Rules	Yes	Yes	No
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	virtualized (EC2) 10Gbps network
Sort rate	**1.42 TB/min**	**4.27 TB/min**	**4.27 TB/min**
Sort rate/node	**0.67 GB/min**	**20.7 GB/min**	**22.5 GB/min**

Apache Spark - Architecture



Spark applications run as independent sets of processes on a cluster, coordinated by the `SparkContext` object in your main program

Spark RDDs – DataFrames - SparkSQL

At its core, Spark operates on the concept of Resilient Distributed Datasets, or RDD's:

- Resilient - if data in memory is lost, it can be recreated
- Distributed - immutable distributed collection of objects in memory partitioned across many data nodes in a cluster
- Dataset - initial data can from from files, be created programmatically, from data in memory, or from another RDD

DataFrames API is a data abstraction framework that organizes your data into named columns:

- Create a schema for the data
- Conceptually equivalent to a table in a relational database
- Can be constructed from many sources including structured data files, tables in Hive, external databases, or existing RDDs
- Provides a relational view of the data for easy SQL like data manipulations and aggregations
- Under the hood, it is a row of RDD's

SparkSQL is a Spark module for structured data processing. You can interact with SparkSQL through:

- SQL
- DataFrames API
- Datasets API

Spark RDDs – DataFrames - SparkSQL

At its core, Spark operates on the concept of Resilient Distributed Datasets, or RDD's:

- Resilient - if data in memory is lost, it can be recreated
- Distributed - immutable distributed collection of objects in memory partitioned across many data nodes in a cluster
- Dataset - initial data can from from files, be created programmatically, from data in memory, or from another RDD

Distributed Array of Data

DataFrames API is a data abstraction framework that organizes your data into named columns:

- Create a schema for the data
- Conceptually equivalent to a table in a relational database
- Can be constructed from many sources including structured data files, tables in Hive, external databases, or existing RDDs
- Provides a relational view of the data for easy SQL like data manipulations and aggregations
- Under the hood, it is a row of RDD's

RDD with Schema

SparkSQL is a Spark module for structured data processing. You can interact with SparkSQL through:

- SQL
- DataFrames API
- Datasets API

RDD + Schema + SQL

Let's do it !!!

Setup – Environment and Software

1. Apache Spark
2. Hadoop Yarn, Mesos, Standalone (for development/testing)
3. Scala/Python and SBT
4. IntelliJ/Any-IDE
5. **Command line tools**

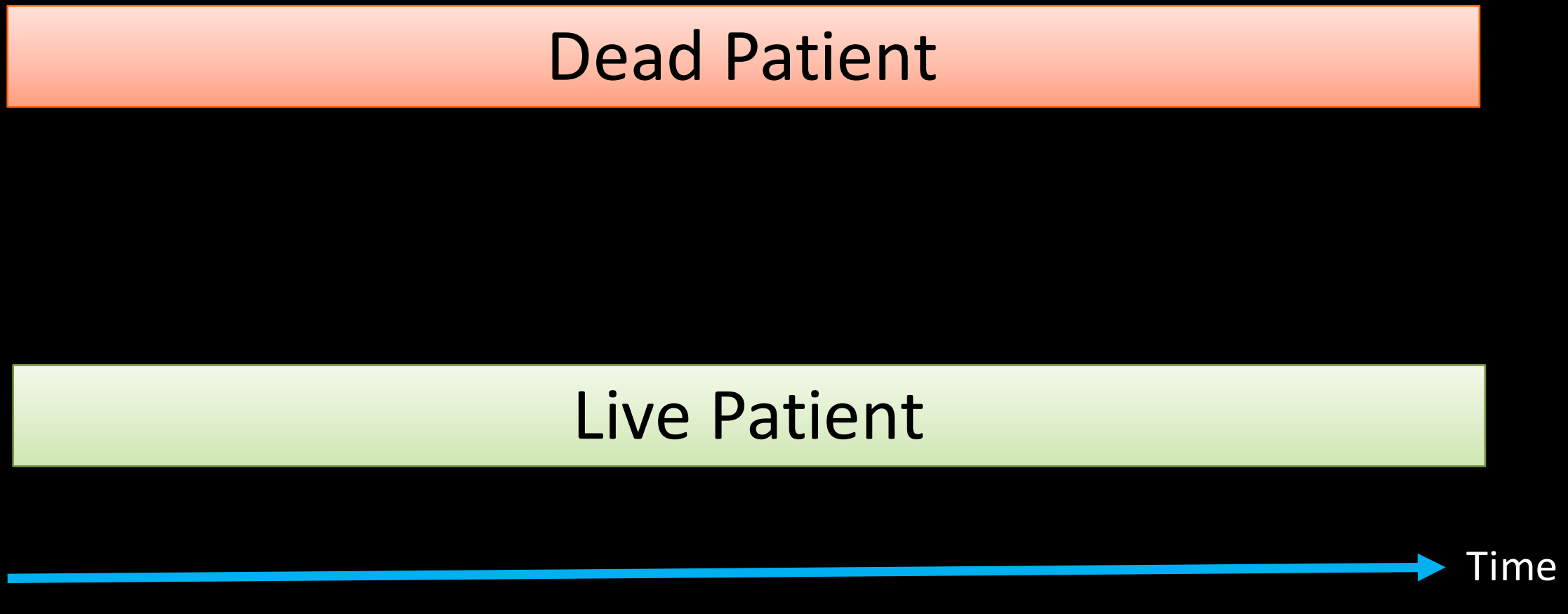
Lets see some code

where is domain?

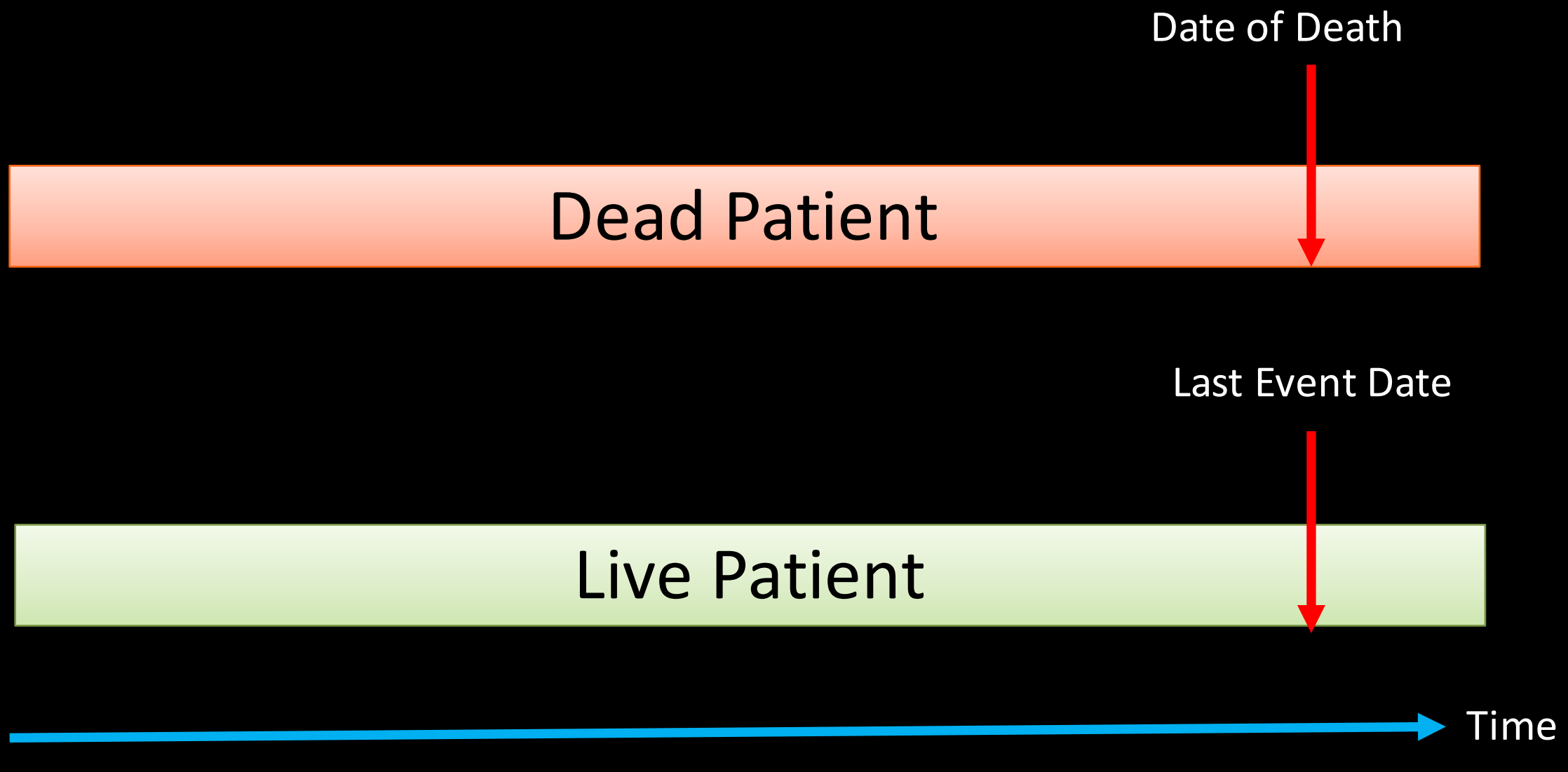
Predict Patient mortality

1. Patient events data
    1. Drug
    2. Lab result
    3. Diagnosis code
2. Patient's mortality info

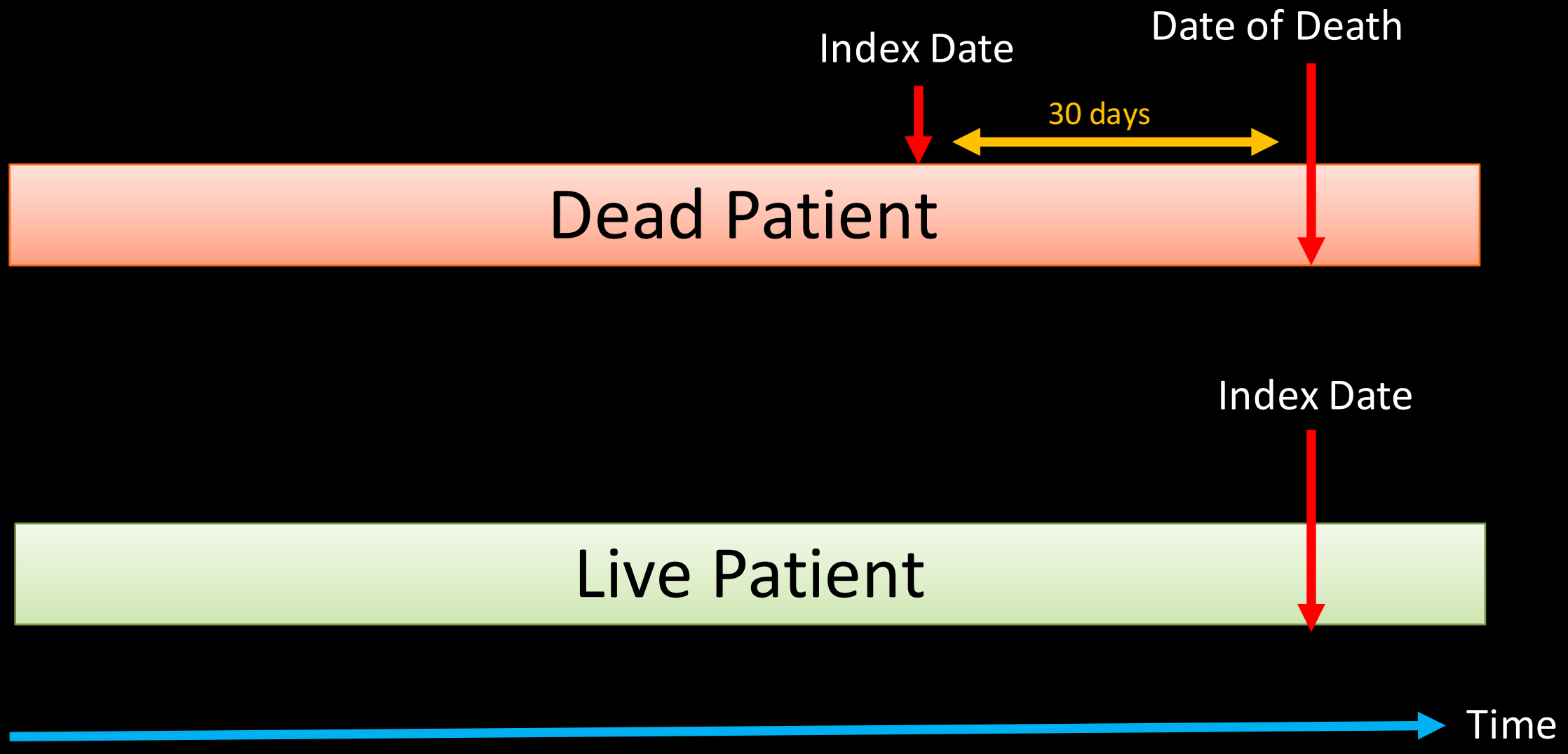
Index date and Prediction Window



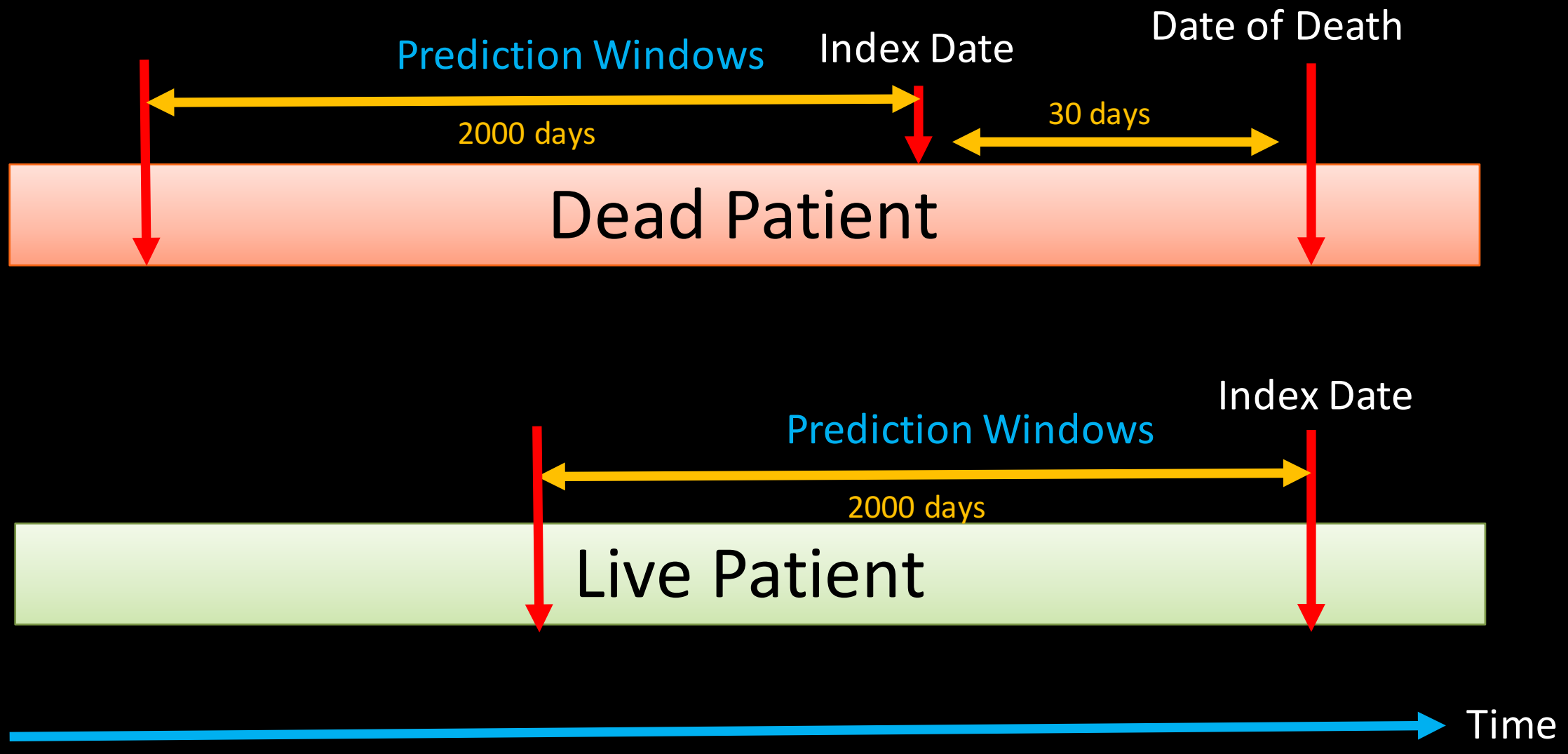
Index date and Prediction Window



Index date and Prediction Window



Index date and Prediction Window



Thank You

Email: joychak1@[yahoo/gmail].com