CALLING ALL DEVELOPERS!

Introduction to machine Learning

Apache Cassandra™
Apache Spark™

DataStax

Director of Developer Relations

clunven

clunven

clun

- Trainer
- Public Speaker
- Developers Support
- Developer Applications
- Developer Tooling

- Creator of ff4j (ff4j.org)
- Maintainer for 8 years+

- Happy developer for 14 years
- Spring Petclinic Reactive & Starters
- Implementing APIs for 8 years

Cédrick Lunven

Developer Advocate

@stefano-lottini

@hemidactylus

@rsprrs

- Developer/Architect
- Apache Cassandra™ certified

- Background in computational physics
- Distributed systems
- Love to teach and communicate

dtsx.io/stefano

Stefano Lottini

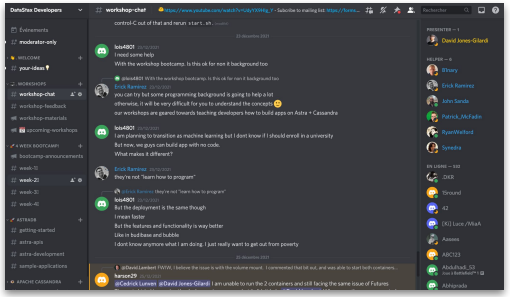
DataStax Developers Crew

Livestream: youtube.com/DataStaxDevs

YouTube

Twitch

Questions: https://dtsx.io/discord

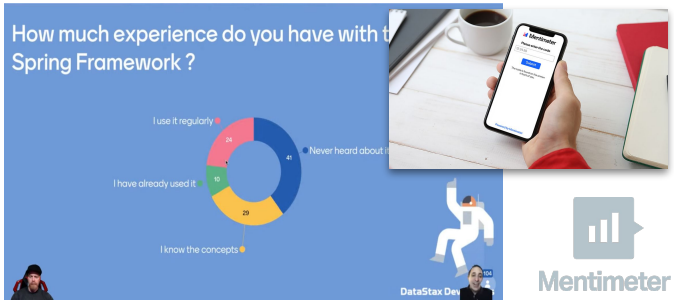


YouTube
(with nightbot)

Discord
(#workshop-chat)

!discord

Games and quizzes: menti.com



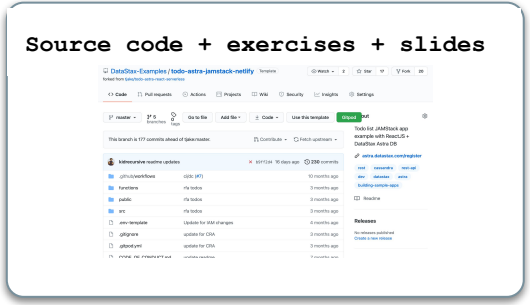
Mentimeter

!menti

Live Sessions

Nothing to install !

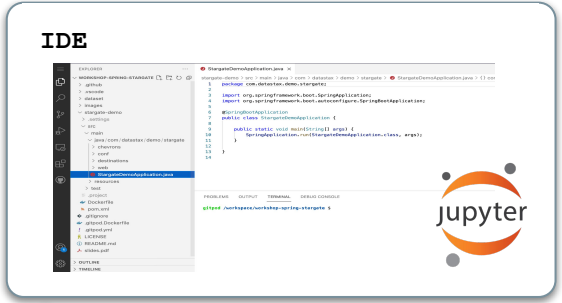
Source code + exercises + slides



GitHub

!github

IDE



Gitpod

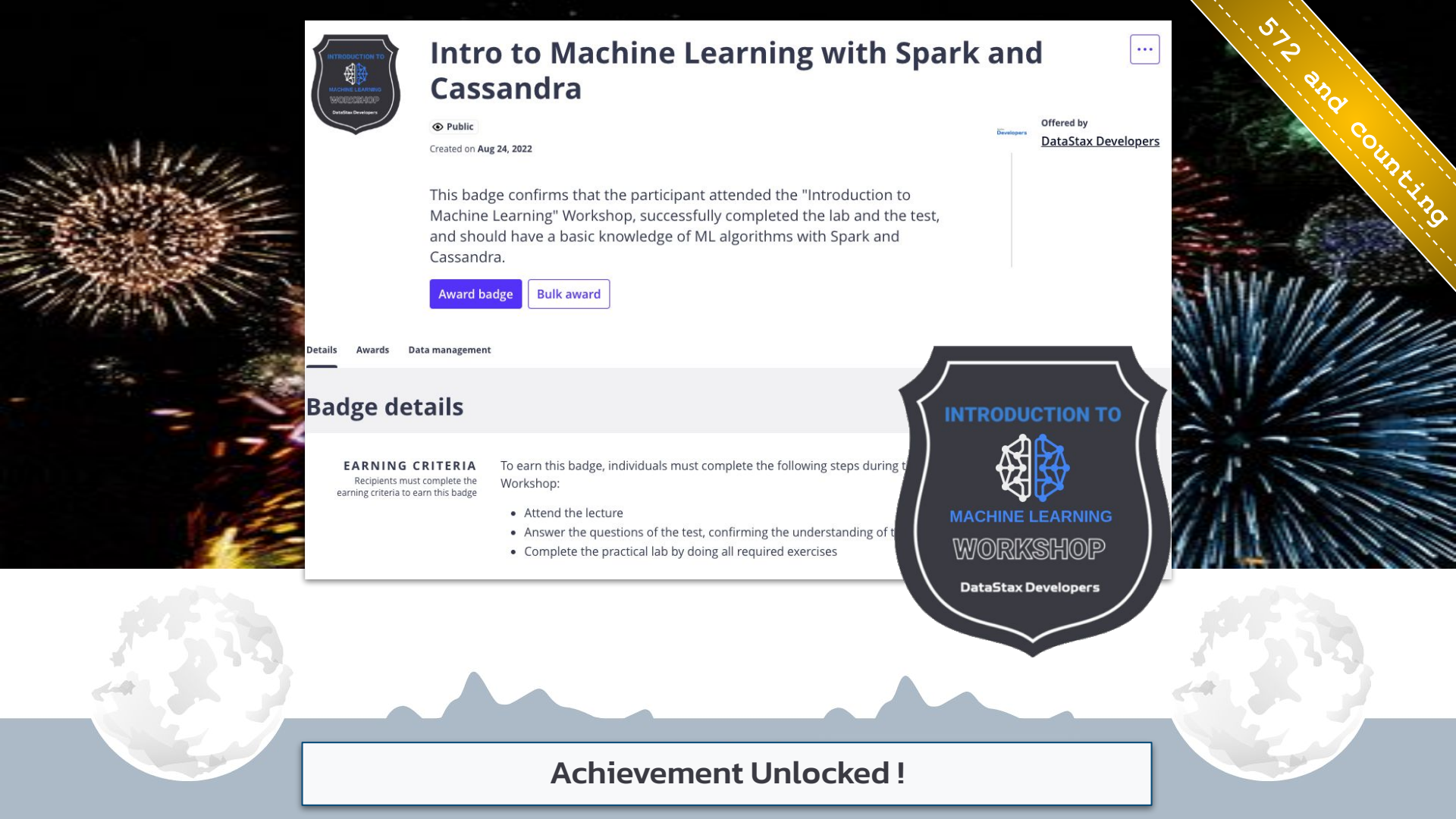
!gitpod

Database + Api + Streaming

DataStax Astra DB

!astra

Hands-On Housekeeping


572 and counting
Intro to Machine Learning with Spark and Cassandra
Public
Created on Aug 24, 2022
Offered by
DataStax Developers
This badge confirms that the participant attended the "Introduction to Machine Learning" Workshop, successfully completed the lab and the test, and should have a basic knowledge of ML algorithms with Spark and Cassandra.
Award badge
Bulk award
Details
Awards
Data management
Badge details
EARNING CRITERIA
Recipients must complete the earning criteria to earn this badge
To earn this badge, individuals must complete the following steps during t
Workshop:
Attend the lecture
Answer the questions of the test, confirming the understanding of t
Complete the practical lab by doing all required exercises
INTRODUCTION TO
MACHINE LEARNING
WORKSHOP
DataStax Developers
Achievement Unlocked !

01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

05

Algos 2
Classification
Recommendation

06

What's next?
Quiz, Homework, Next week

Agenda

01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

05

Algos 2
Classification
Recommendation

06

What's next?
Quiz, Homework, Next week

Agenda

Machine Learning

Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

Wikipedia.org

Machine learning is the scientific study of algorithms a
models that computer systems use to perform
using explicit instructions, relying on p is
seen as a subset of artificial int
build a mathematical m
data", in order t
program explicitly
in gorithms are used
v filtering and computer
o develop a conventional
alg ming the task.

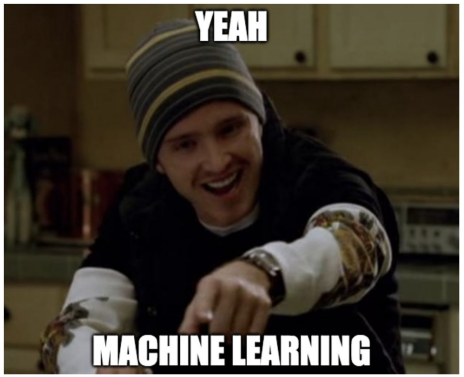
Wikipedia.org

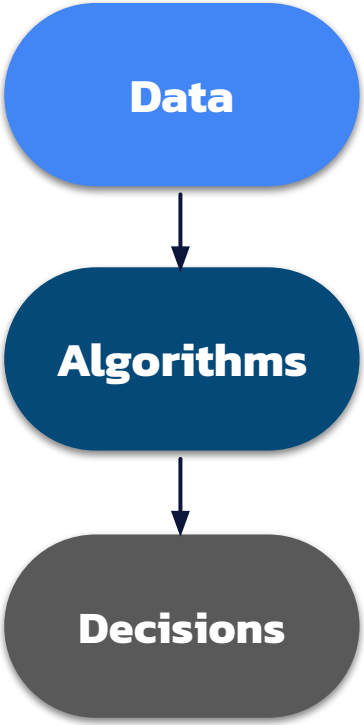
"Machine Learning is a science of drawing circles [and colorizing them]"

A. Volochnev

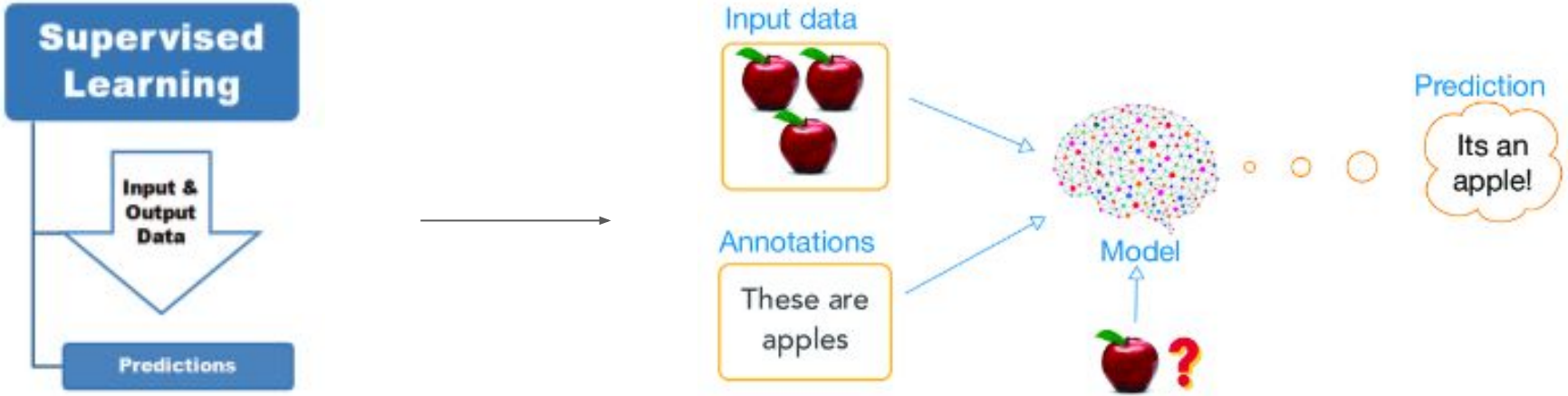
Machine Learning is a scientific way to process raw data using algorithms to make better decisions.

No magic, just billions rows of data and two buckets of mathematics. Voilà!

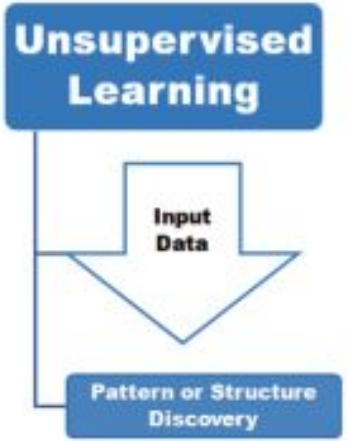


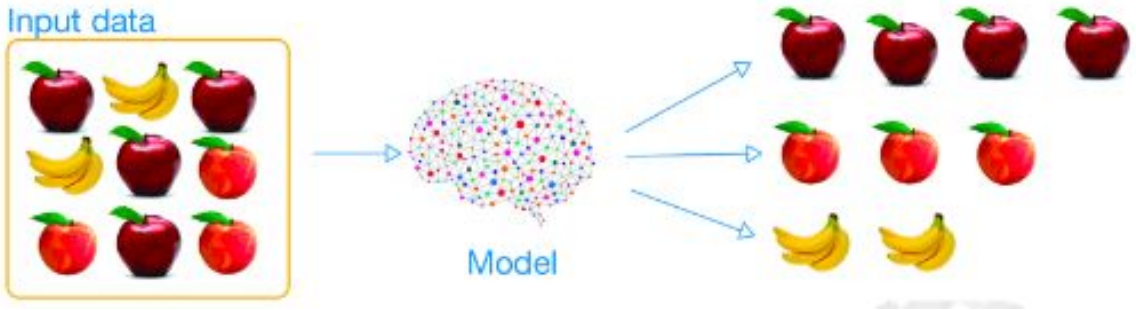


How it works ?



- **Knowledge of the output**: learn with expert
  - Data are **labelled** with class or value
  - **Goal**: predict the class



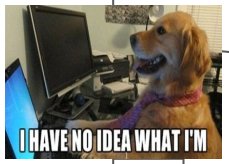


- **No Knowledge of the output**: self-guided
  - Data are not **labelled** with class or value
  - **Goal**: Determine Patterns of Grouping

- FP-Growth
- K-Means Clustering
- Naive Bayes
- Decision Trees
- Neural Networks
- Collaborative Filtering
- Logistic Regression
- Support Vector Machines
- Linear Regression

- Apriori Algorithm
- Case-based Reasoning
- Dimensionality Reduction Algorithms
- Gradient Boosting Algorithms
- Hidden Markov Models
- Self-organizing Map
- K-Nearest Neighbour
- ECLAT

And still counting...



Pre Categorized Data

Supervised Learning

Predictive Models

- Classification → Naive Bayes
- Regressions → Random Forest

Semi Supervised

Unlabelled data

Unsupervised Learning

Pattern Recognition

- Clustering → K-means
- Association → FP-Growth
- Dimensionality Reduction → Collaborative Filtering

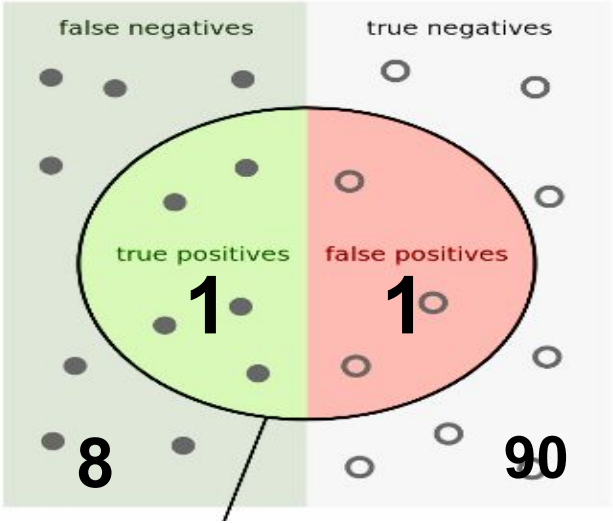
Reinforcement Learning

Machine Learning for today

Data Model
“Quality”

100 people, 9 have malignant tumor (very bad), 91 have benign tumor (bad)

True Positive (TP): • Reality: Malignant • ML model predicted: Malignant • **Number of TP results: 1**	**False Positive (FP):** • Reality: Benign • ML model predicted: Malignant • **Number of FP results: 1**
False Negative (FN): • Reality: Malignant • ML model predicted: Benign • **Number of FN results: 8**	**True Negative (TN):** • Reality: Benign • ML model predicted: Benign • **Number of TN results: 90**



Example and images from:
https://shiffdag.medium.com/what-is-accuracy-precision-and-recall-and-why-are-they-important-ebfcb5a10df2

Evaluating data model

Accuracy is an evaluating classification models metric, it is the fraction of predictions model identified correctly.

$$\text{Accuracy} = \frac{\text{Correct Prediction (TP + TN)}}{\sum \text{Predictions}}$$

True Positive (TP): • Reality: Malignant • ML model predicted: Malignant • **Number of TP results: 1**	**False Positive (FP):** • Reality: Benign • ML model predicted: Malignant • **Number of FP results: 1**
False Negative (FN): • Reality: Malignant • ML model predicted: Benign • **Number of FN results: 8**	**True Negative (TN):** • Reality: Benign • ML model predicted: Benign • **Number of TN results: 90**

What is the accuracy here ?

How many go home without proper treatment ?

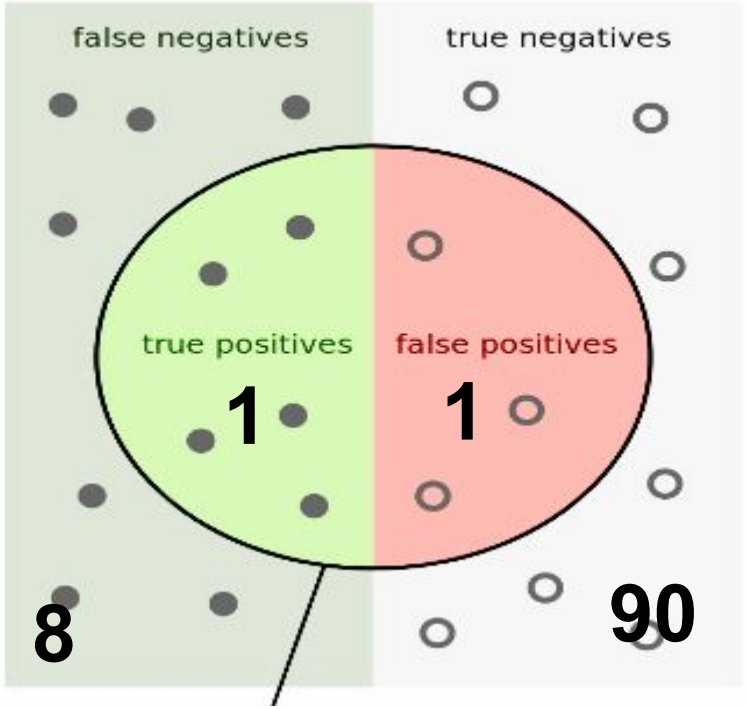
True Positive (TP): • Reality: Malignant • ML model predicted: Malignant • **Number of TP results: 1**	**False Positive (FP):** • Reality: Benign • ML model predicted: Malignant • **Number of FP results: 1**
False Negative (FN): • Reality: Malignant • ML model predicted: Benign • **Number of FN results: 8**	**True Negative (TN):** • Reality: Benign • ML model predicted: Benign • **Number of TN results: 90**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{1 + 90}{1 + 90 + 1 + 8} = 0.91$$

Accuracy

Precision counts true positives out of all true and false positives.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\sum \text{Positives (TP + FP)}}$$



What is the precision here?

$$\text{Precision} = \frac{\text{green}}{\text{green + red}}$$

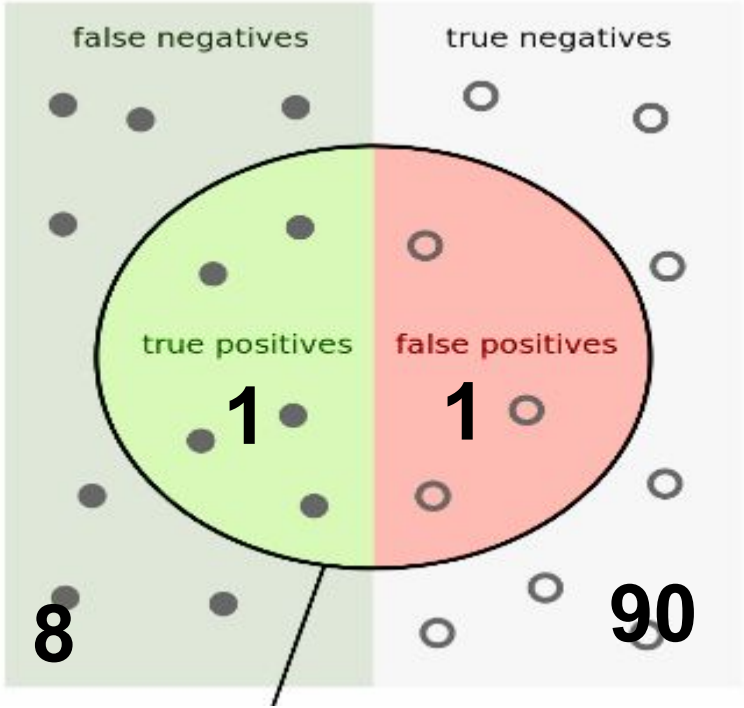
True Positives (TPs): 1	False Positives (FPs): 1
False Negatives (FNs): 8	True Negatives (TNs): 90

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{1}{1 + 1} = 0.5$$

Precision

Recall correctly identified positives out of all real positives.

$$\text{Prediction} = \frac{\text{True Positives (TP)}}{\sum \text{Correct (TP + FN)}}$$



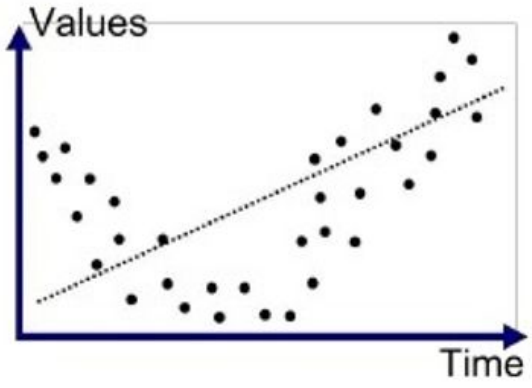
What is the recall here?

$$\text{Recall} = \frac{\text{green}}{\text{green}}$$

Let's calculate recall for our tumor classifier:

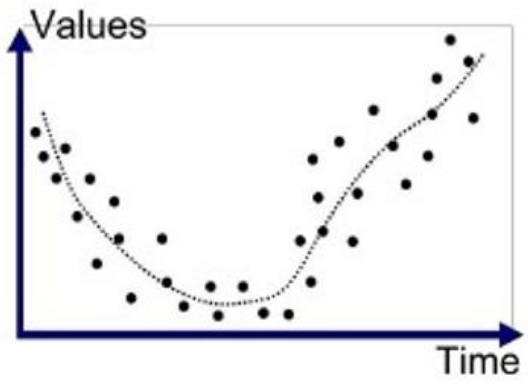
True Positives (TPs): 1	False Positives (FPs): 1
False Negatives (FNs): 8	True Negatives (TNs): 90

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{1}{1 + 8} = 0.11$$



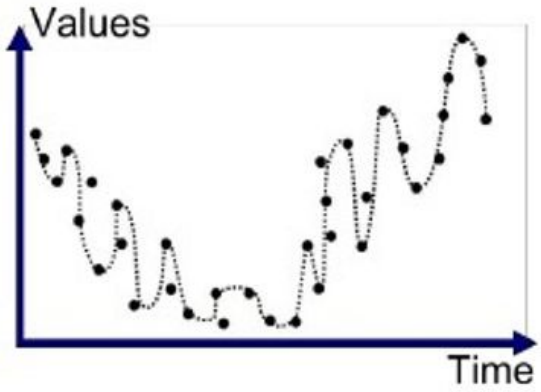
Underfitted

Not accurate, too simple



Good Fit/Robust

Good, well generalised



Overfitted

Over-trained, perfect on train data, fails on test data

Under fitted vs over-fitted model

01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

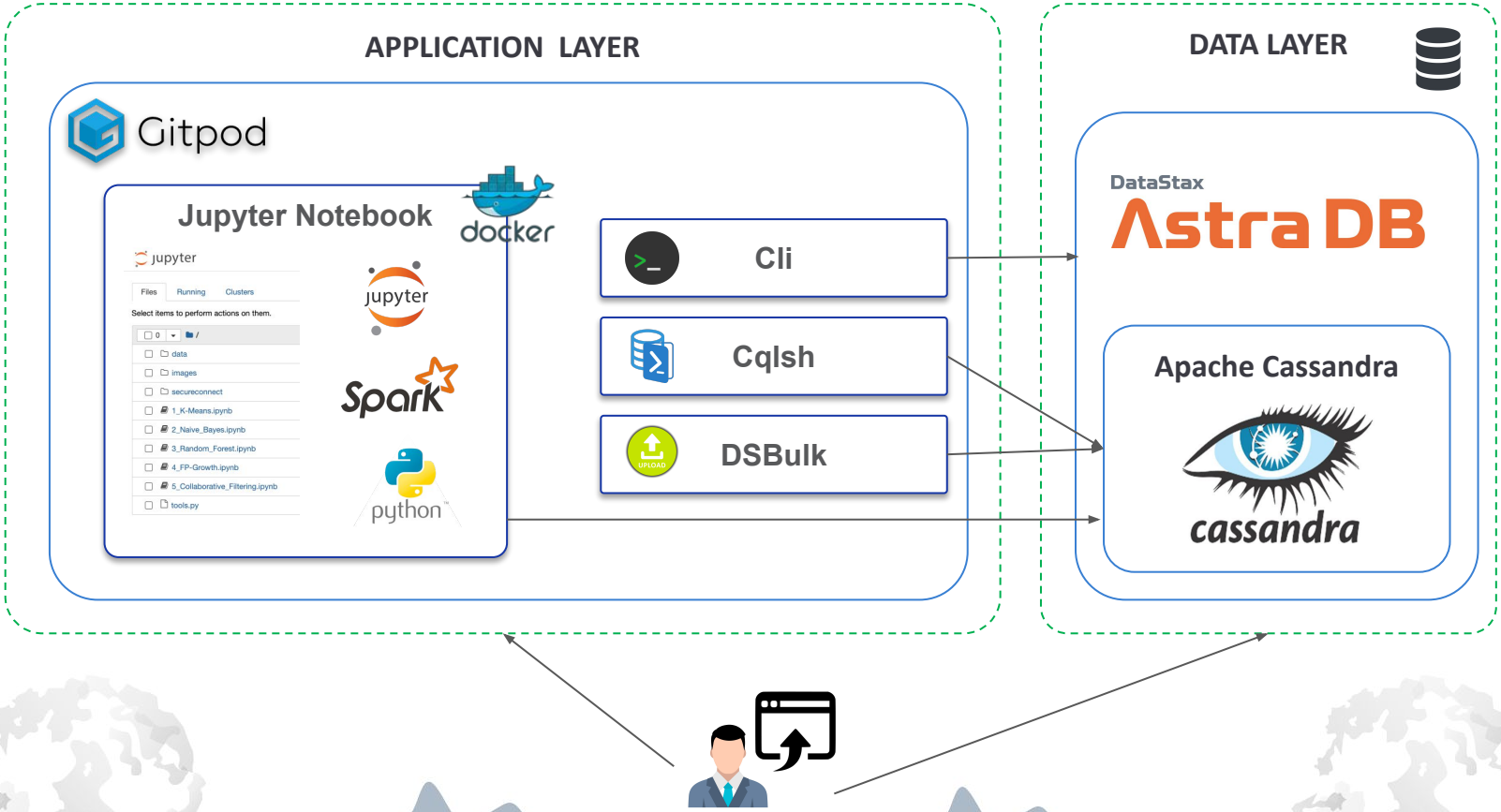
05

Algos 2
Classification
Recommendation

06

What's next?
Quiz, Homework, Next week

Agenda



Know your Tools

APPLICATION LAYER

DATA LAYER

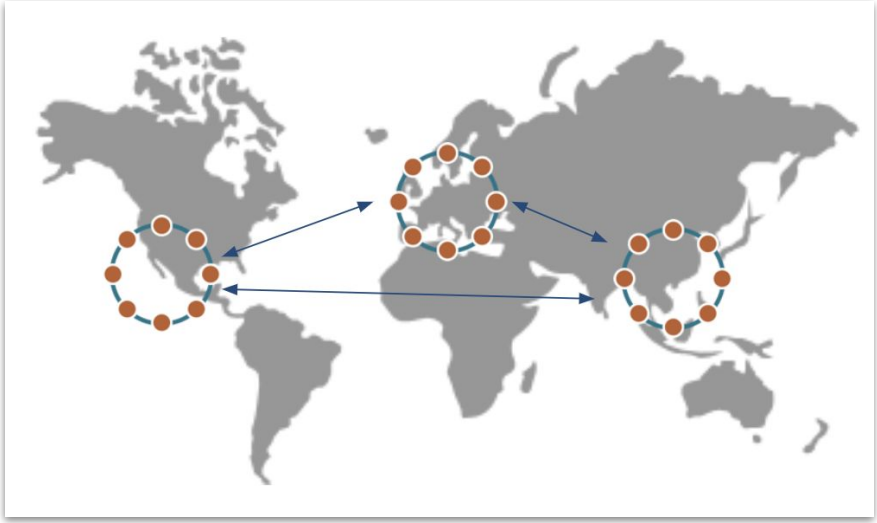
DataStax Astra DB

Apache Cassandra

cassandra

Know your Tools

- Big Data Ready
- Read / Write Performance
- Linear Scalability
- Highest Availability
- Self-Healing and Automation
- Geographical Distribution
- Platform Agnostic
- Vendor Independent

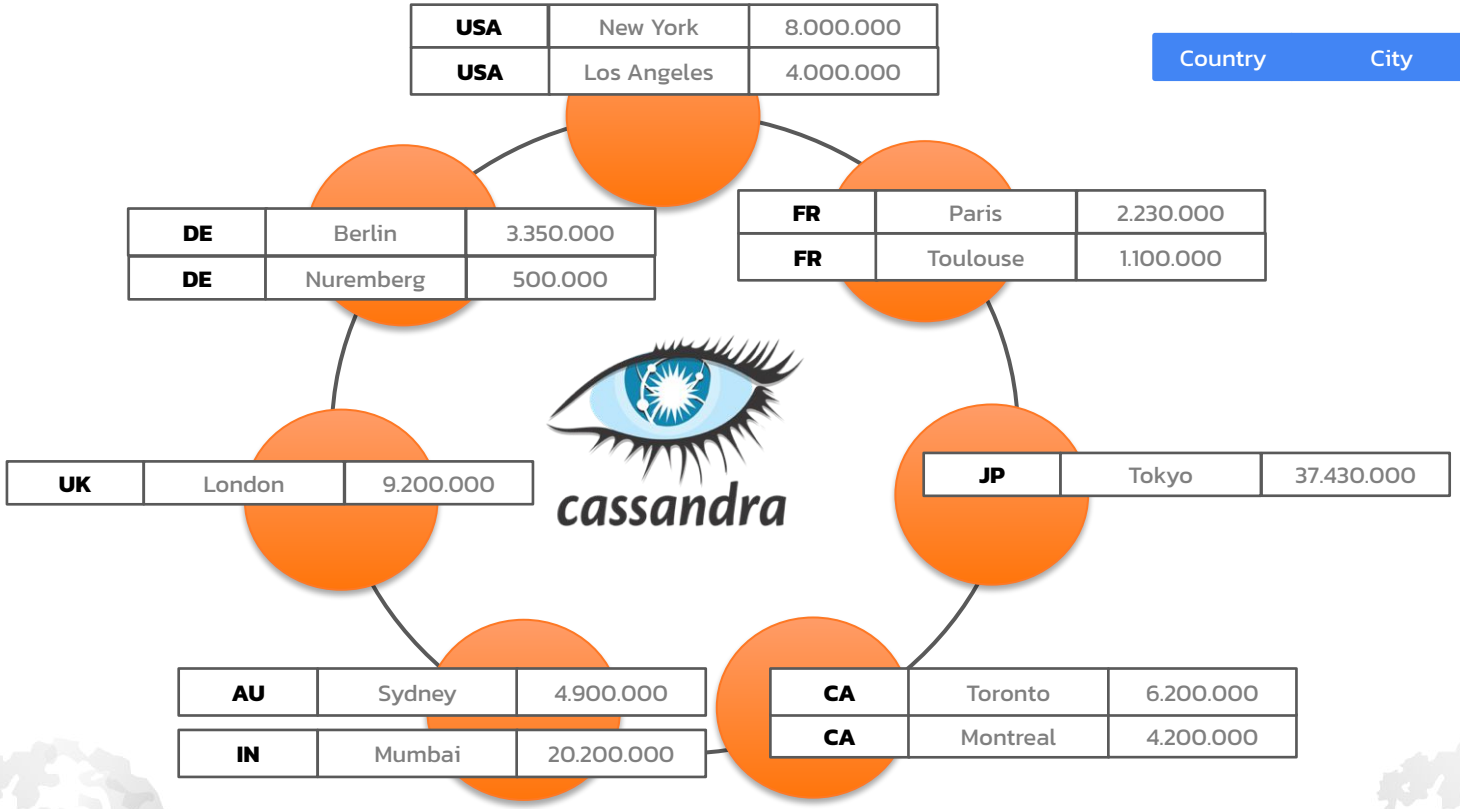


Apache Cassandra™

Country	City	Population
USA	New York	8.000.000
USA	Los Angeles	4.000.000
FR	Paris	2.230.000
DE	Berlin	3.350.000
UK	London	9.200.000
AU	Sydney	4.900.000
DE	Nuremberg	500.000
CA	Toronto	6.200.000
CA	Montreal	4.200.000
FR	Toulouse	1.100.000
JP	Tokyo	37.430.000
IN	Mumbai	20.200.000

Partition Key

Data is Distributed



Country	City	Population

Data is Distributed

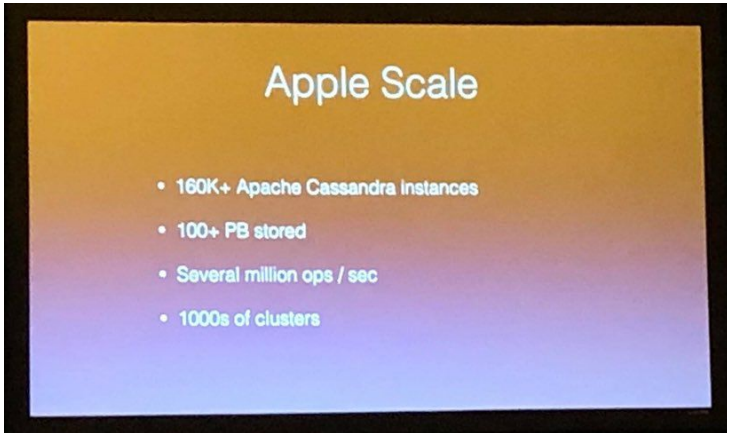
Apache Cassandra @ Netflix

- 98% of streaming data is stored in Apache Cassandra
- Data ranges from customer details to viewing history to billing and payments
- Foundational datastore for serving millions of operations per second

- 30 million ops/sec on most active single cluster
- 500 TB most dense single cluster
- 9216 CPUs in biggest cluster

O(100)	Clusters
O(10000)	Instances
O(10,000,000)	Replications per second
O(100,000,000)	Operations per second
O(1,000,000,000,000,000)	Petabytes of data

dtsx.io/cassandra-at-netflix



And many others...

Cassandra Biggest Users (and Developers)

Free to Use

Up to 80GB storage and/or 20 million operations monthly.

Serverless

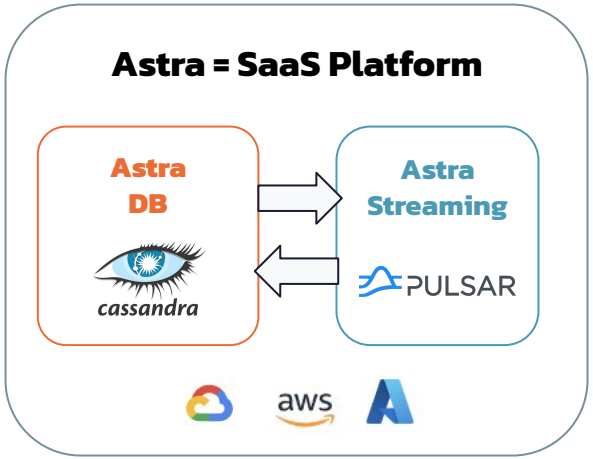
Lower your costs by running Cassandra clusters only when needed.

No Operations

Eliminate the overhead to install, operate, and scale Cassandra.

Data APIs

Work natively with Document (JSON), REST, GraphQL and gRPC APIs.



Global Scale

Put your data where you need it without compromising performance, availability or accessibility.

End-to-End Security

Secure connect with VPC peering and Private Link. Bring your own encryption key management. SAML SSO secure account access.

Zero Lock-in

Deploy on AWS, GCP or Azure and keep compatibility with open-source Cassandra.

Relational Indexes

Storage Attached Indexing (SAI) lets you query tables using any columns.

Astra = Cassandra As a Service++

Lab 1

Create Your Astra DB Instance

https://github.com/datastaxdevs/workshop-introduction-to-machine-learning#4-create-your-astra-db-instance

01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

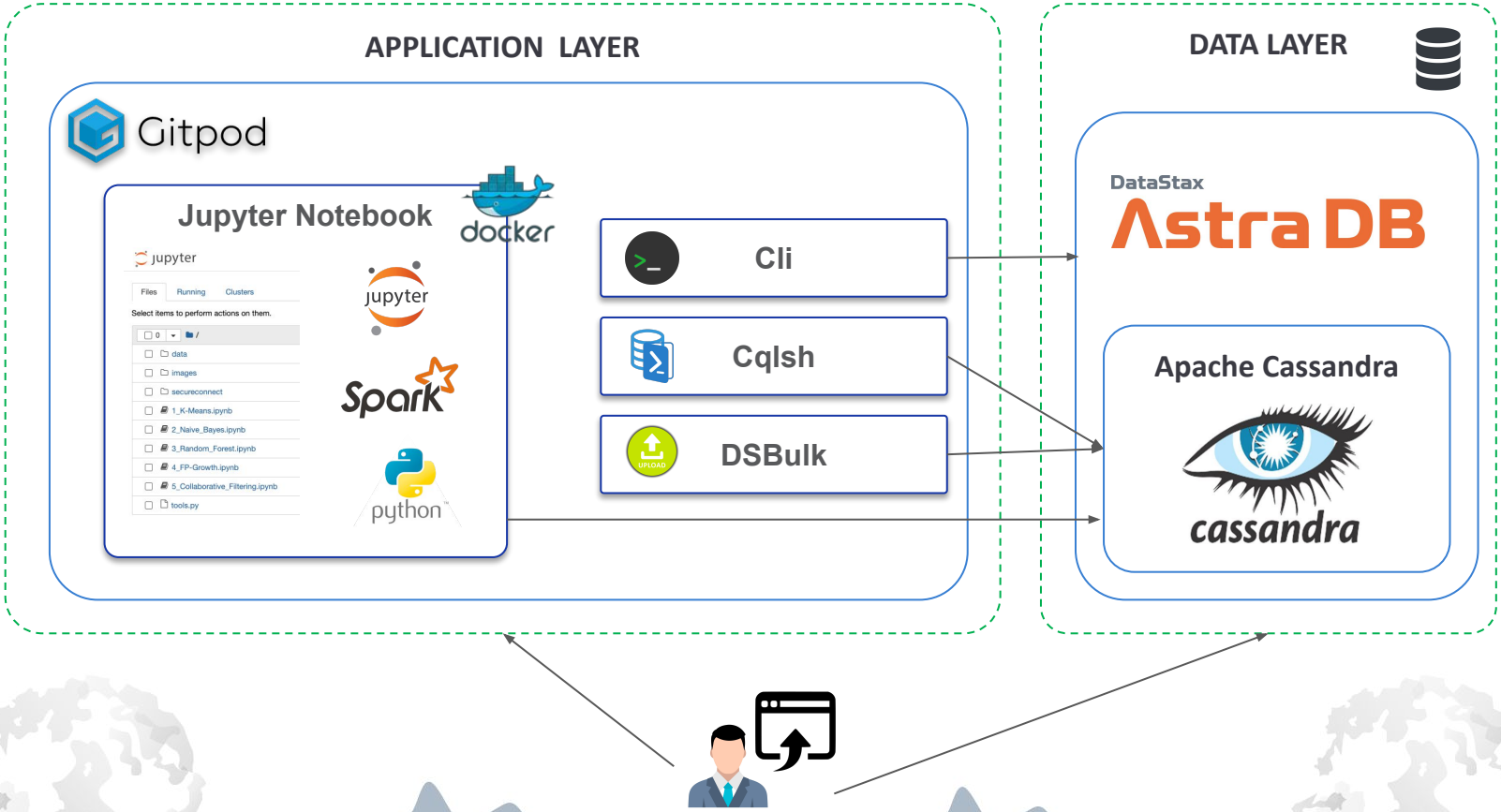
05

Algos 2
Classification
Recommendation

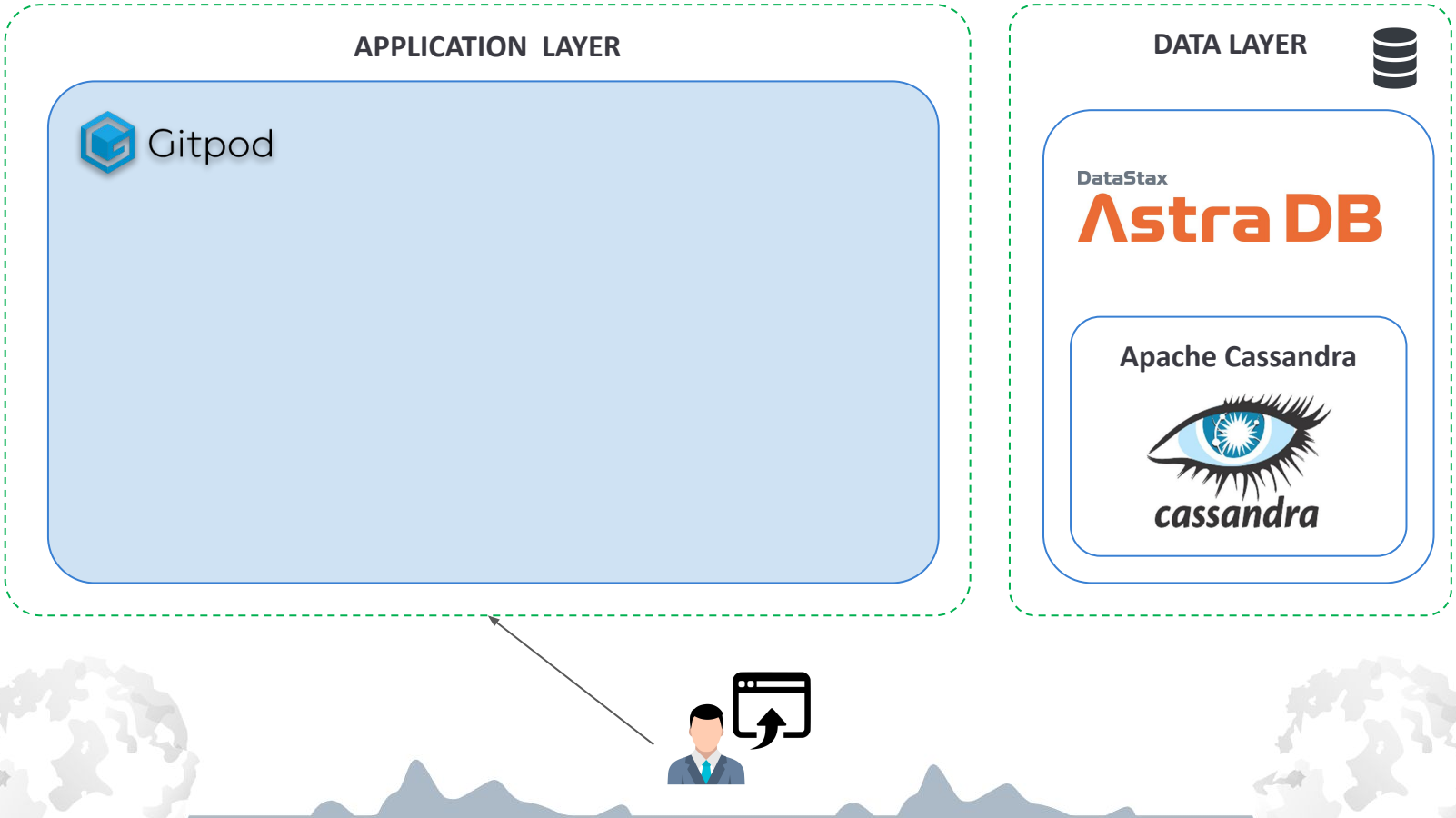
06

What's next?
Quiz, Homework, Next week

Agenda



Know your Tools


APPLICATION LAYER
Gitpod
DATA LAYER
DataStax
Astra DB
Apache Cassandra
cassandra
Gitpod

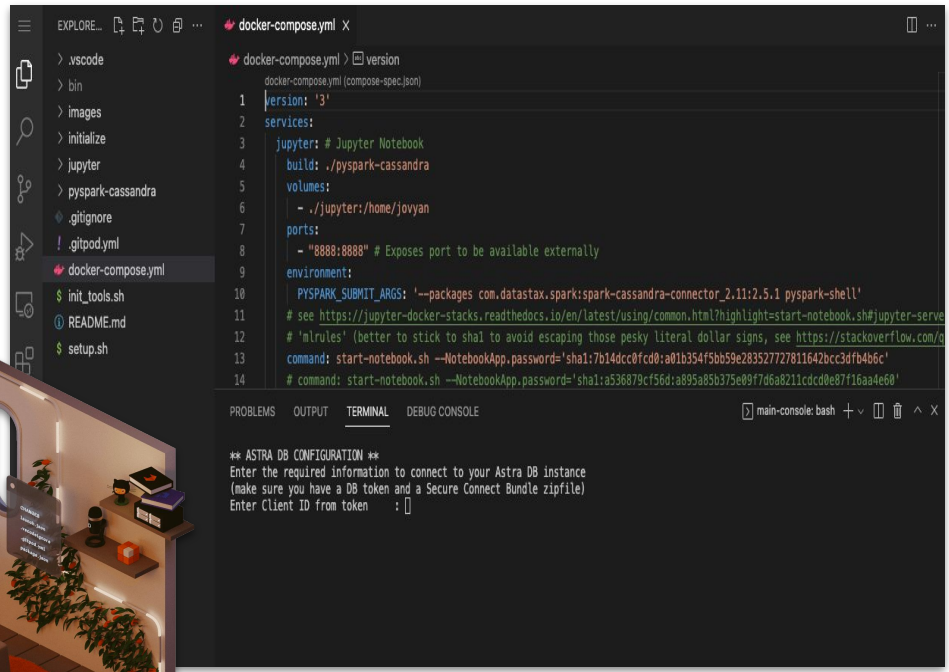
Always ready to code.

Spin up fresh, automated dev environments for each task, in the cloud, in seconds.

Try Now

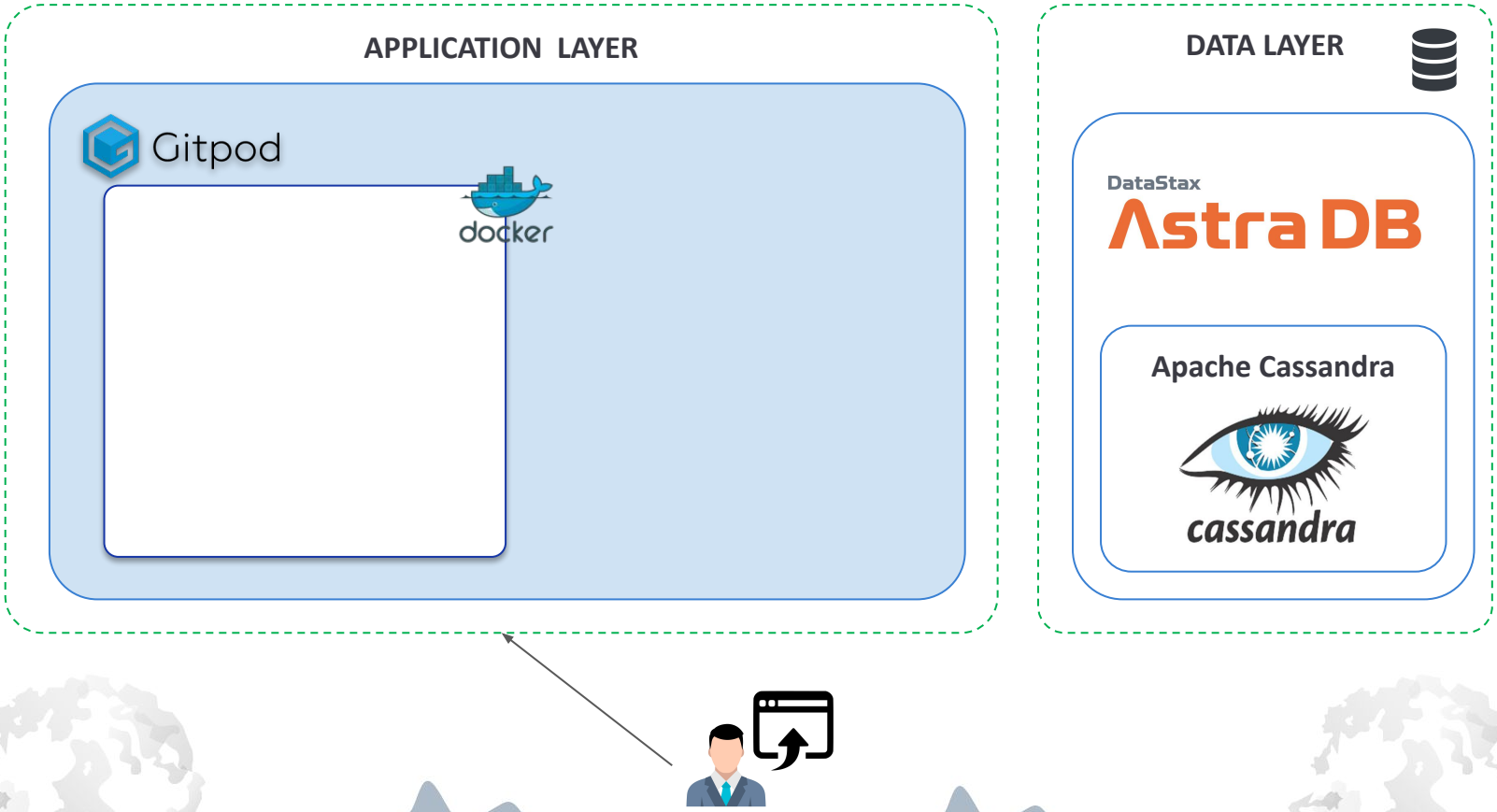
Open a workspace.
Start from any Git context.

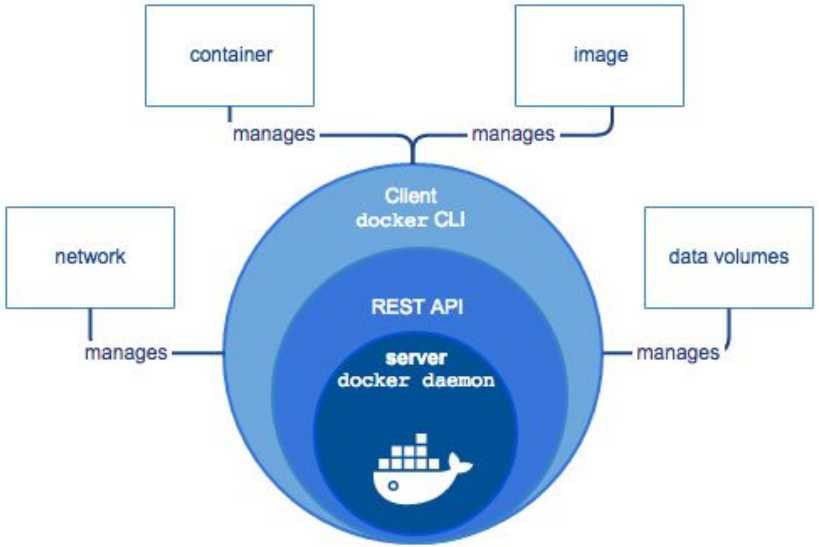
☆ Star 8,829



Gitpod

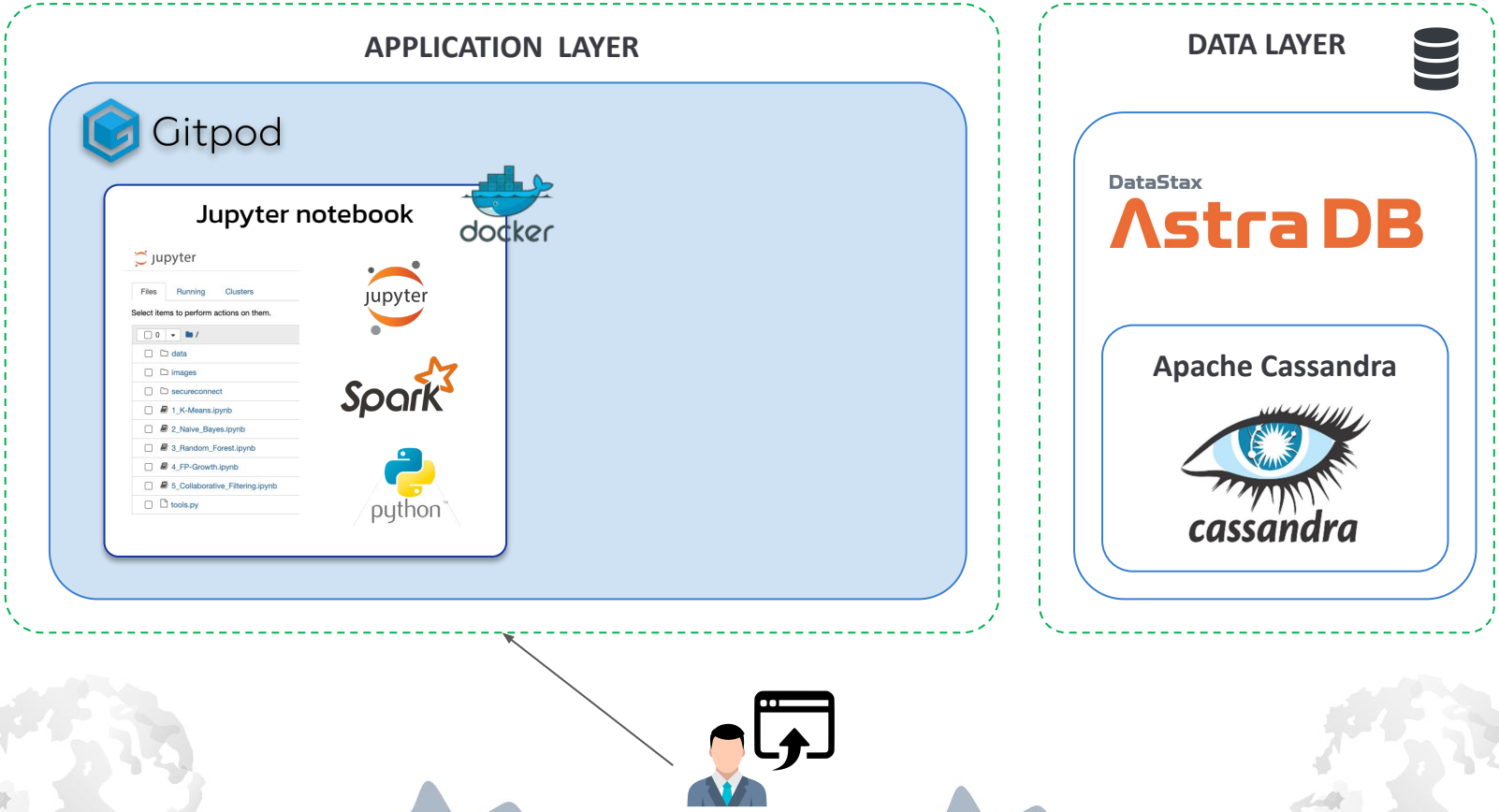
Gitpod


APPLICATION LAYER
Gitpod
docker
DATA LAYER
DataStax
Astra DB
Apache Cassandra
cassandra
Gitpod



Docker Engine is Client-server application with :

- A server which is a type of long-running program called a daemon process (the dockerd command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the docker command)



Jupyter Notebook

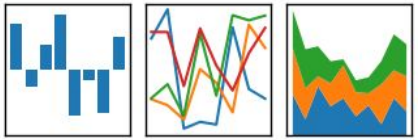
Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.


```
fileName = 'data/ratings.csv'
input_file = open(fileName, 'r')

for line in input_file:
    row = line.split(',')

    query = "INSERT INTO movieratings (userid, movieid, rating, timestamp)"
    query = query + " VALUES (%s, %s, %s, %s)"
    session.execute(query, (int(row[0]), int(row[1]), float(row[2]), row[3]))
```

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



```
In [1]: df = pd.DataFrame({'AAA': [4, 5, 6, 7],
   ...:                    'BBB': [10, 20, 30, 40],
   ...:                    'CCC': [100, 50, -30, -50]})
   ...:

In [2]: df
Out[2]:
   AAA  BBB  CCC
0    4   10  100
1    5   20   50
2    6   30  -30
3    7   40  -50
```

Panda

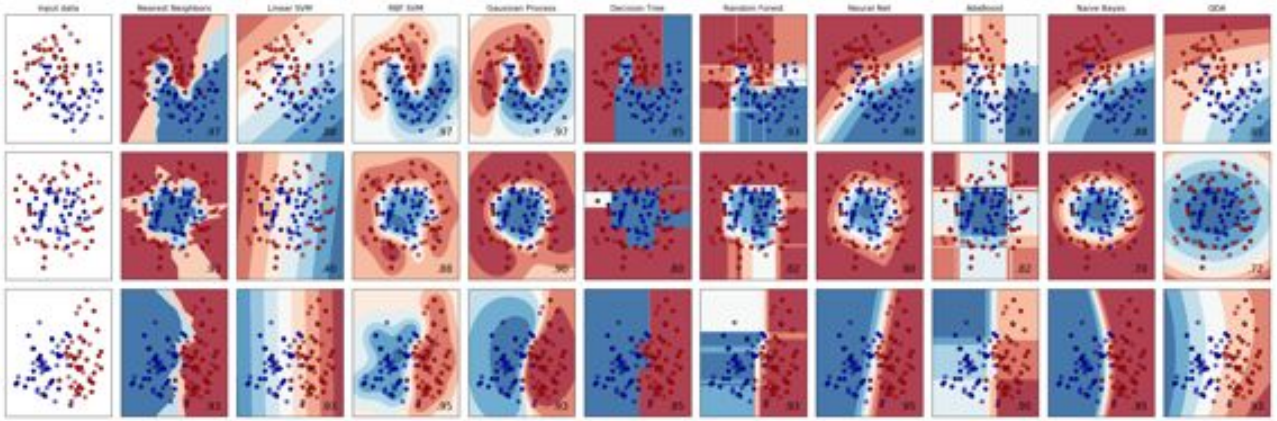
Apache Spark is written in Scala programming language. PySpark has been released in order to support the collaboration of Apache Spark and Python, it actually is a Python API for Spark. In addition, PySpark, helps you interface with Resilient Distributed Datasets (RDDs) in Apache Spark and Python programming language.

NumPy is the fundamental package for scientific computing with Python.

It contains among other things: a powerful N-dimensional array object, sophisticated functions, useful linear algebra, Fourier transform, and random number capabilities.

```
>>> x = np.array([('Rex', 9, 81.0), ('Fido', 3, 27.0)],
...              dtype=[('name', 'U10'), ('age', 'i4'), ('weight', 'f4')])
>>> x
array([('Rex', 9, 81.), ('Fido', 3, 27.)],
      dtype=[('name', 'U10'), ('age', '<i4'), ('weight', '<f4')])
```

An open source, simple and efficient tool for predictive data analysis, accessible to everybody, and reusable in various contexts. Built on NumPy, SciPy, and matplotlib.



01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

05

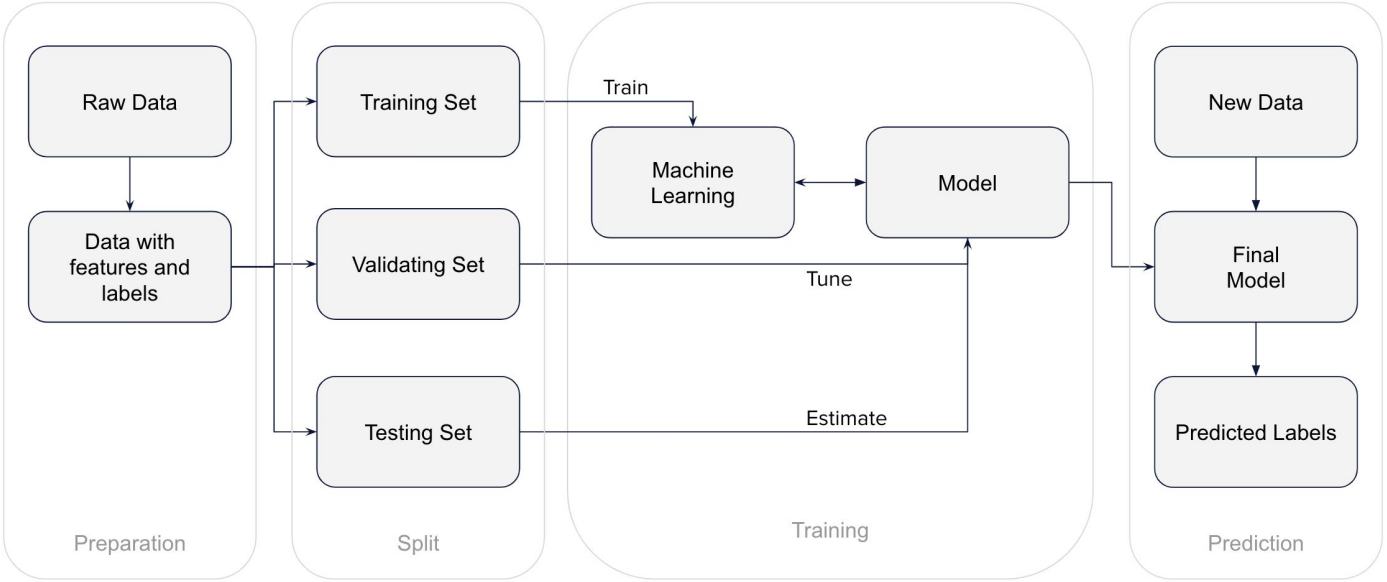
Algos 2
Classification
Recommendation

06

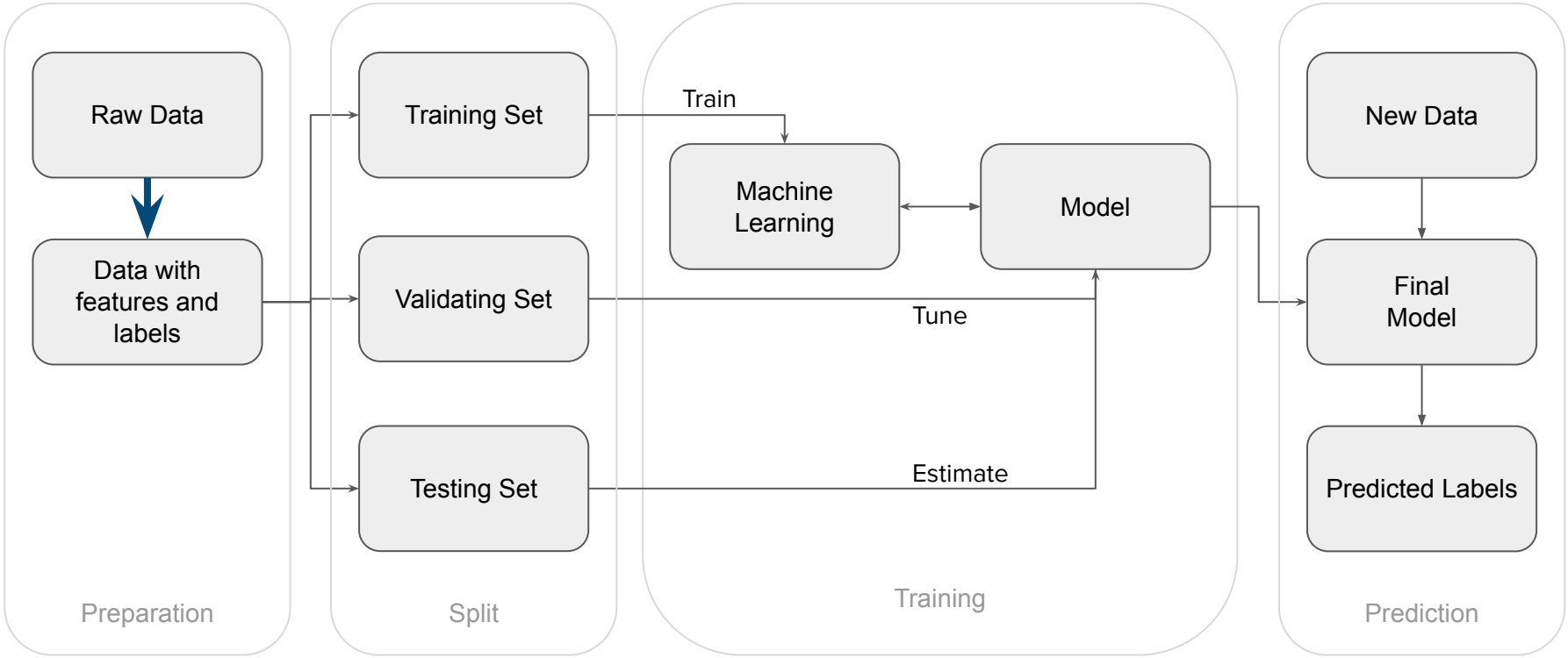
What's next?
Quiz, Homework, Next week

Agenda

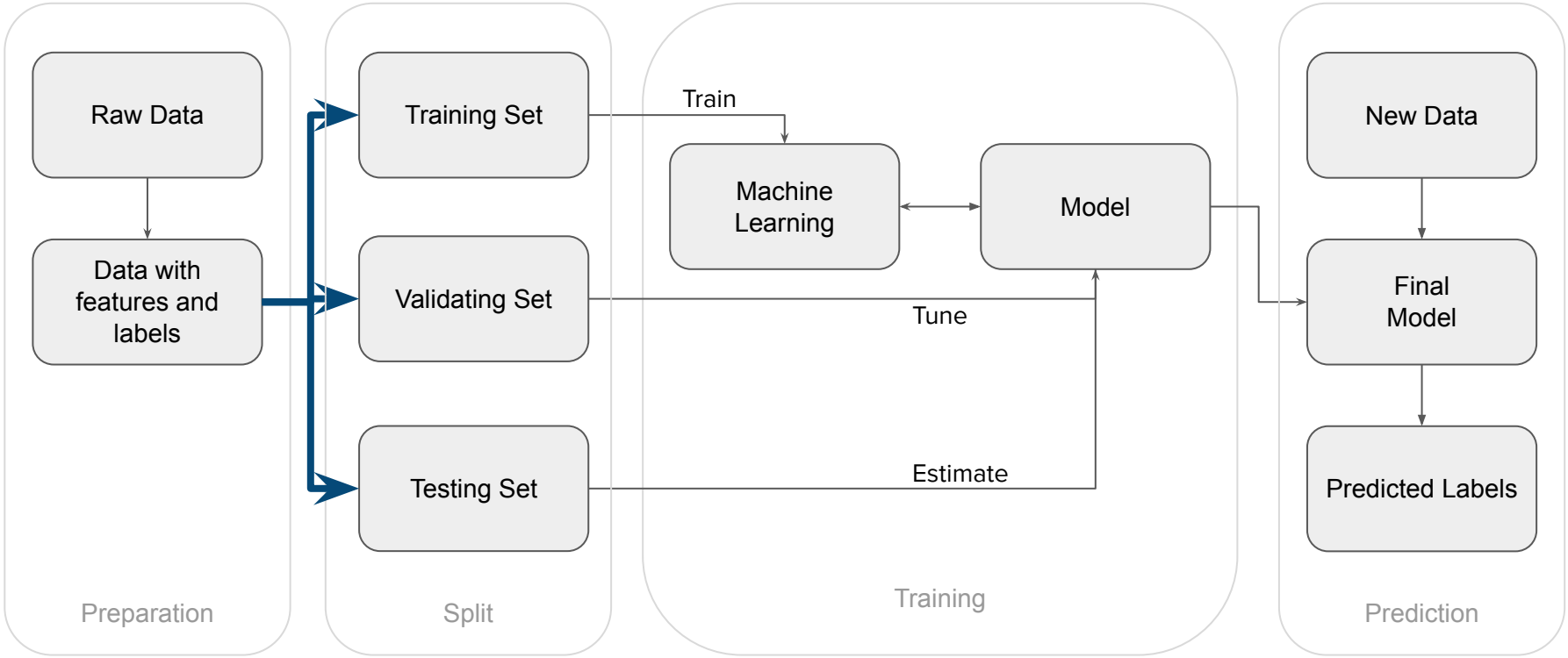
- Question / Hypothesis
- Algorithm Selection
- Data Preparation
- Data Split
- Training
- Tuning
- Testing
- Analysis
- Repeat



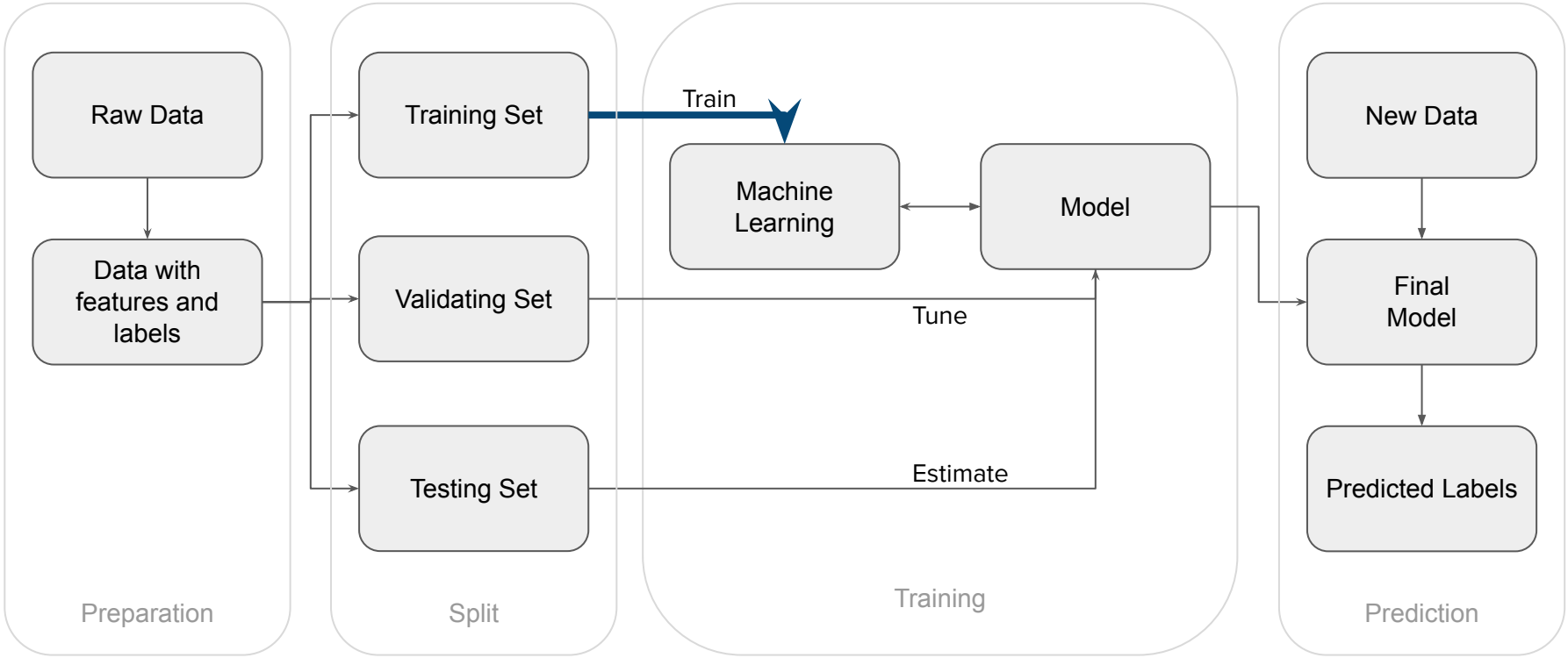
Learning Workflow



Data Preparation


Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

Data Split

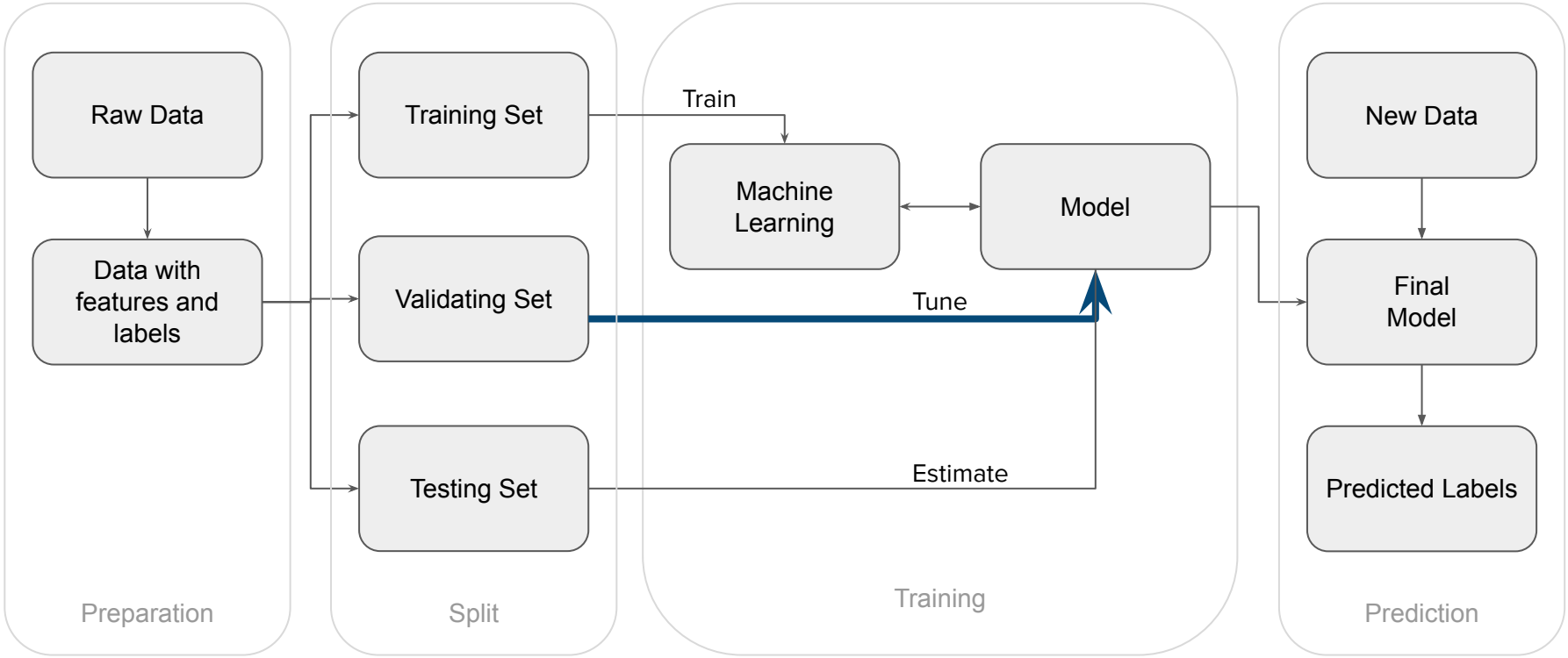

Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

Training

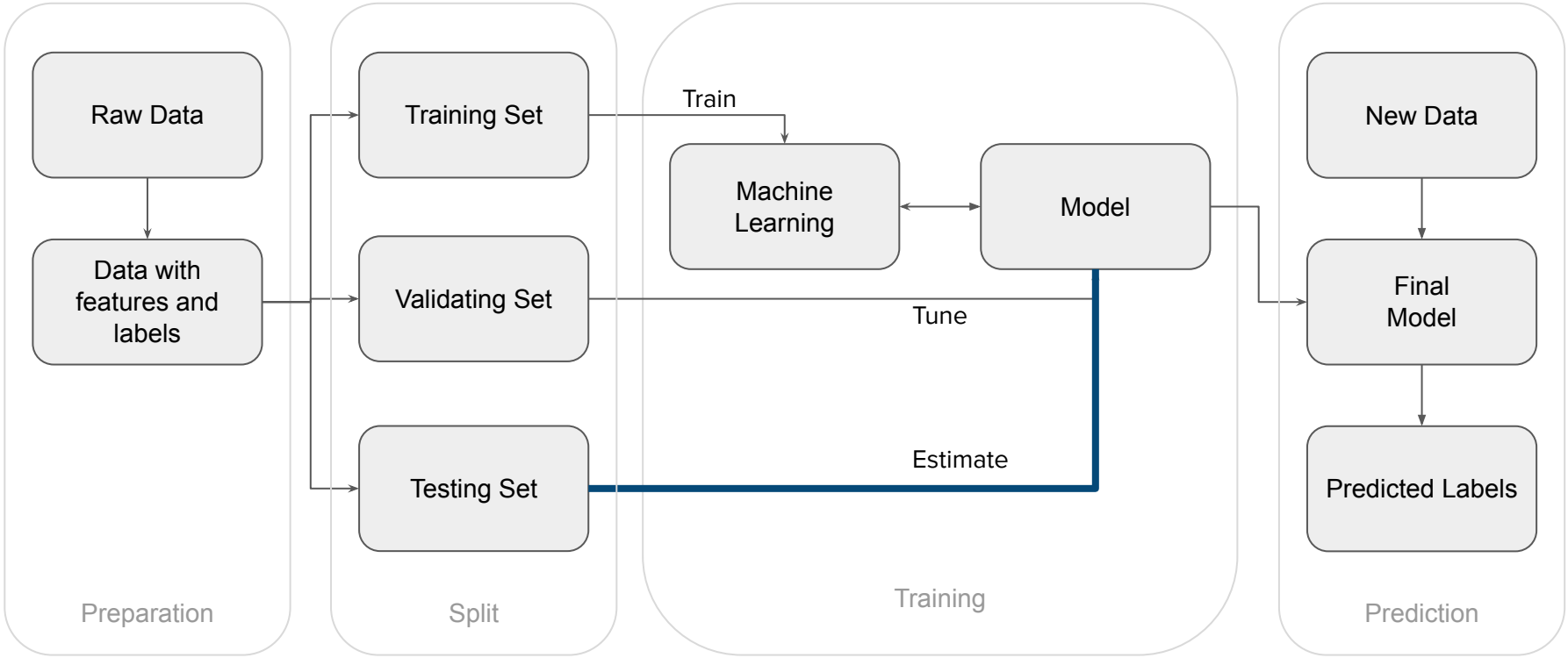
FUN FACT: this image was created by … an algorithm,
starting from the textual prompt: **"a metallic cyborg in a gym"**

`https://huggingface.co/spaces/stabilityai/stable-diffusion`

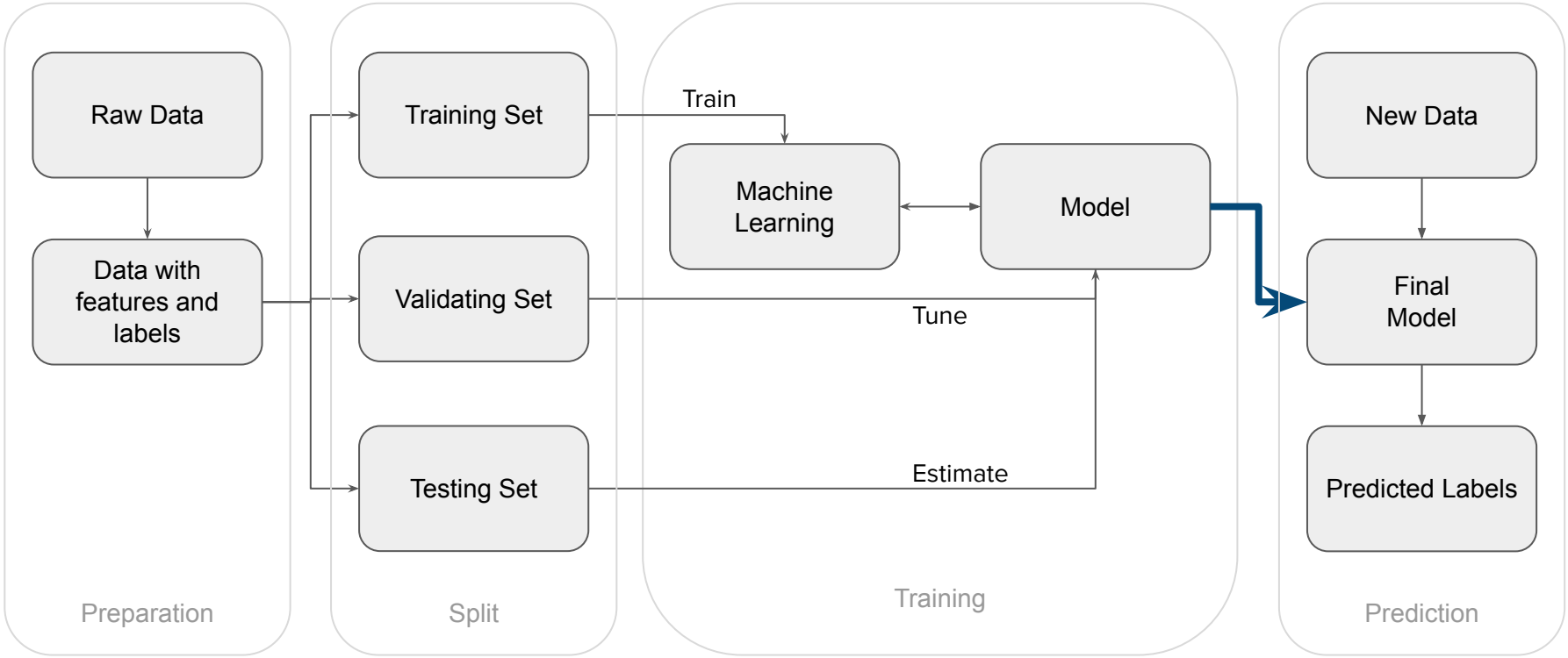
Intermezzo: "training the machine"


Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

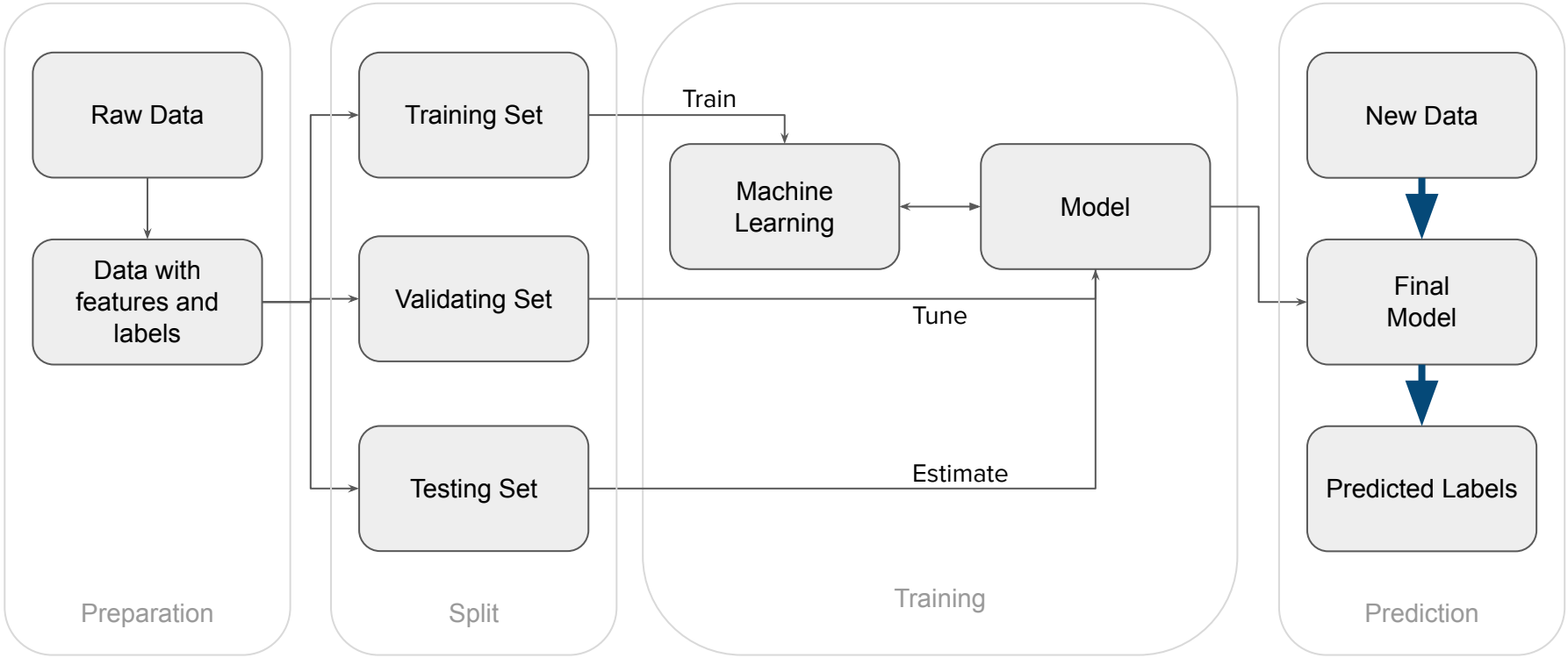
Tuning


Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

Testing


Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

Final Model


Raw Data
Data with features and labels
Training Set
Validating Set
Testing Set
Train
Machine Learning
Model
Tune
Estimate
New Data
Final Model
Predicted Labels
Preparation
Split
Training
Prediction

Prediction

Demo with K-means

http://user.ceng.metu.edu.tr/~akifakkus/courses/ceng574/k-means/

Lab 2

K Means

https://github.com/datastaxdevs/workshop-introduction-to-machine-learning#6-algorithms

Naïve Bayes

Supervised Classification Algorithm

"Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong "naïve" independence assumptions between the features.

Naïve Bayes is a popular method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis."

$$p(D \mid S) = \prod_i p(w_i \mid S)$$

and

$$p(D \mid \neg S) = \prod_i p(w_i \mid \neg S)$$

Using the Bayesian result above, we can write:

$$p(S \mid D) = \frac{p(S)}{p(D)} \prod_i p(w_i \mid S)$$

$$p(\neg S \mid D) = \frac{p(\neg S)}{p(D)} \prod_i p(w_i \mid \neg S)$$

Dividing one by the other gives:

$$\frac{p(S \mid D)}{p(\neg S \mid D)} = \frac{p(S) \prod_i p(w_i \mid S)}{p(\neg S) \prod_i p(w_i \mid \neg S)}$$

Which can be re-factored as:

$$\frac{p(S \mid D)}{p(\neg S \mid D)} = \frac{p(S)}{p(\neg S)} \prod_i \frac{p(w_i \mid S)}{p(w_i \mid \neg S)}$$

statistical independency

P(A , X) = P(A) . P(X)

$$P(y|x_1, \ldots, x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

Naïve Bayes Algorithm

Okey, but why Naïve? Because it doesn't consider relations between facts. For example, if I write a word "Happy", the probability of the next word to be "Birthday" is obviously higher than "Funerals". Say "long long ago" and a person next to you most probably will continue: "in a galaxy far far away". This algorithm is called Naïve because it considers every fact as a stand-alone, not related to others.

It seems to be a serious flaw but surprisingly it isn't – on the reasonable amounts of data the NB may outperform many other more sophisticated algorithms. Additionally, it's a great for the parallel computing which makes it lightning fast.

Speaking to Bayes:

- Is it warm outside?
- Yes.
- Is it cold outside?
- No.

Speaking to a Human:

- Is it warm outside?
- Yes.
- Is it cold outside?
- *Are you an idiot!?*

Can we classified an email as SPAM only know the title contains words money and easy ?

Step I: Prepare and label data

Get cash easy!

Win money now!

Greetings from Dad

Could you lend me money?

It was easy!

I miss you

Check your trip itinerary

SPAM	NOT SPAM
• Money transfer received • Get cash easy! • Win money now!	• Greetings from Dad • Could you lend me money? • It was easy! • I miss you • Check your trip itinerary

P(spam | ‘Easy’, ‘Money’)

Naïve Bayes Algorithm

SPAM
• Money transfer received • Get cash easy! • Win money now!
NOT SPAM
• Greetings from Dad • Could you lend me money? • It was easy! • I miss you • Check your trip itinerary

P('Money' | spam) = ⅔
P('Money' | not) = ⅕

P('Easy' | spam) = ⅓
P('Easy' | not) = ⅕

P(spam) = ⅜
P(not) = ⅝

P(spam | 'Easy', 'Money') = P('Easy', 'Money' | spam) * P (spam)
= P('Easy'| spam) * P('Money' | spam) * P (spam)
= ⅓ ⅔ ⅜
= 1/12

P(not | 'Easy', 'Money') = P('Easy', 'Money' | not) * P (not)
= P('Easy'| not) * P('Money' | not) * P (not)
= ⅕ * ⅕ * ⅝
= 1/40

P(spam | 'Easy', 'Money') + P(not | 'Easy', 'Money') = 1

$$P(\text{spam} \mid \text{'Easy'}, \text{'Money'}) = \frac{P(\text{spam} \mid \text{'Easy'}, \text{'Money'})}{P(\text{spam} \mid \text{'Easy'}, \text{'Money'}) + P(\text{not} \mid \text{'Easy'}, \text{'Money'})} = \frac{1/12}{1/12 + 1/40}$$

P(not | 'Easy', 'Money') = 3/13

P(spam | 'Easy', 'Money') = 10/13 = 76.9%

Naive Bayes Computation

Lab 3

Naive Bayes

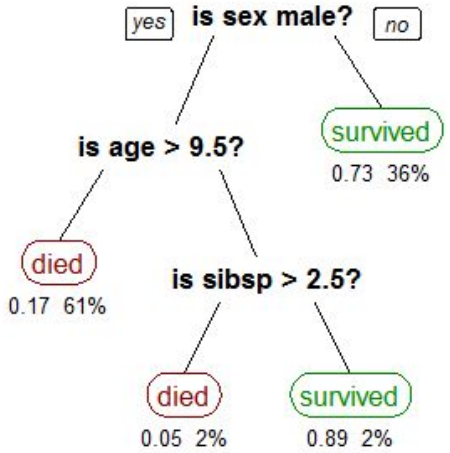
https://github.com/datastaxdevs/workshop-introduction-to-machine-learning#6-algorithms

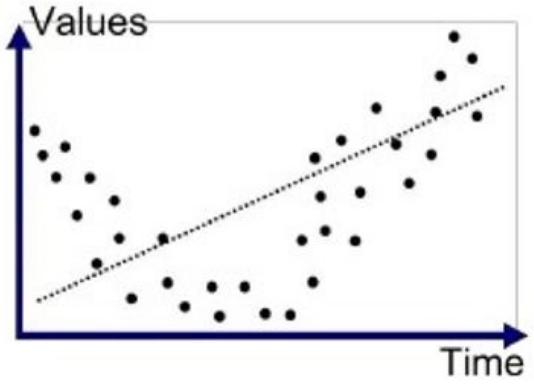
Random Forest

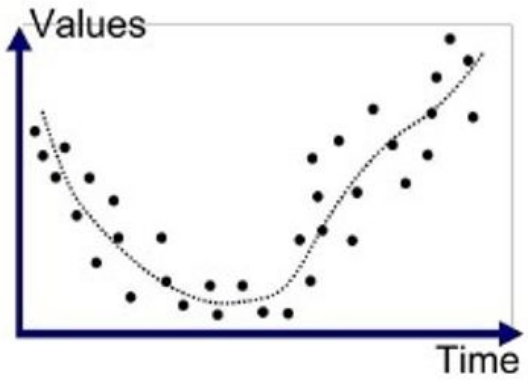
Supervised Classification Ensemble Method

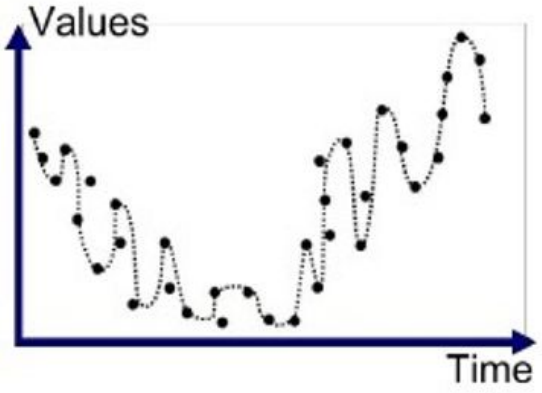
"A decision tree is a simple representation for classifying. Assume that all of the input features have finite discrete domains, and there is a single target feature called the "classification". Each element of the domain of the classification is called a class. A decision tree is a tree in which each internal node is labeled with an input feature. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target or output feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes, signifying that the data set has been classified by the tree into either a specific class, or into a particular probability distribution"

Titanic Survival Decision Tree









Underfitted

Good Fit/Robust

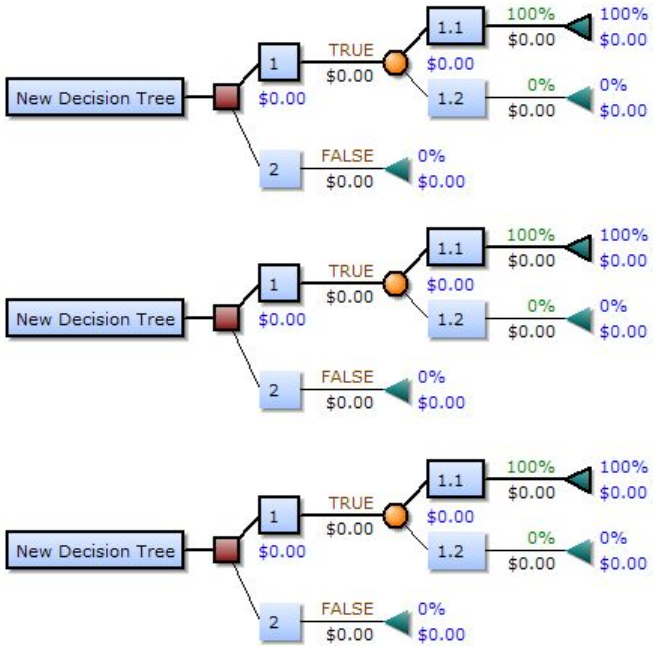
Overfitted

Under-fitter vs over-fitted

Random Forest Ensemble Method

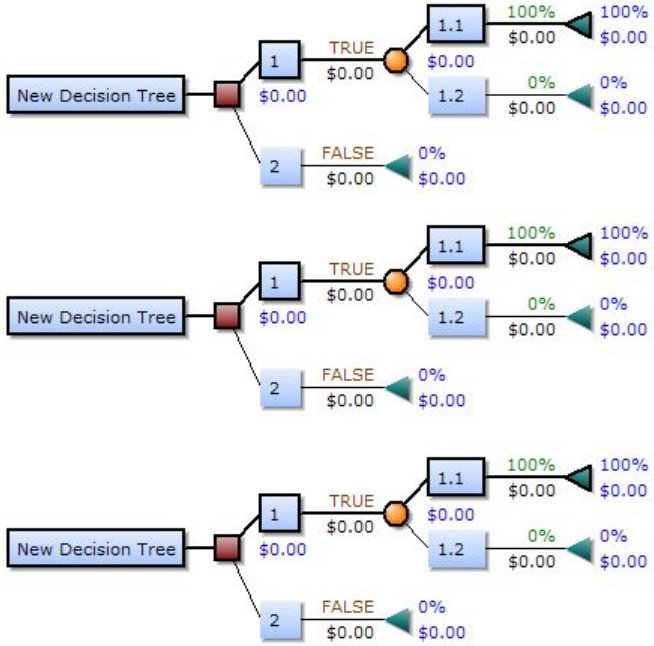
“Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

The first algorithm for random decision forests was created by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.”



"Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

The first algorithm for random decision forests was created by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg."



Random Forest Ensemble

Lab 4

Random Forest

https://github.com/datastaxdevs/workshop-introduction-to-machine-learning#6-algorithms

FP-Growth

Association-rule-based recommendation system

Association-rule learning

A class of models to infer relations between elements in a list of sets. E.g. market-basket analysis:

"Customers who bought x, y, z also bought w"

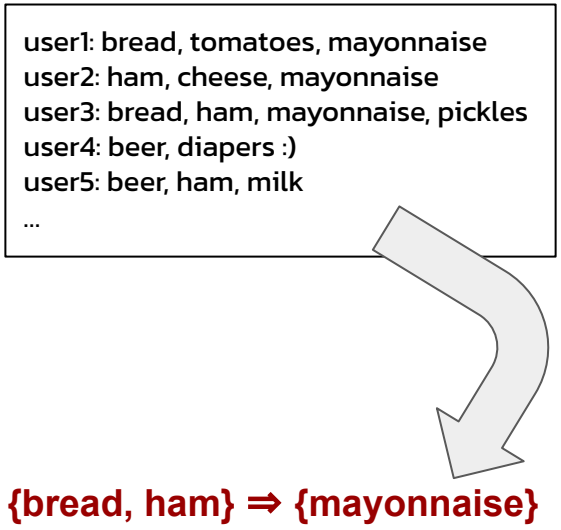
Generally proceed in two steps:

1. identify items that are often together
2. recast this information as "association rules"

FP-Growth

A method to build the "frequent itemsets" in 2 passes

Start from item counts and then rearrange items in a tree (a *trie*): frequent itemsets are found as paths in this tree.

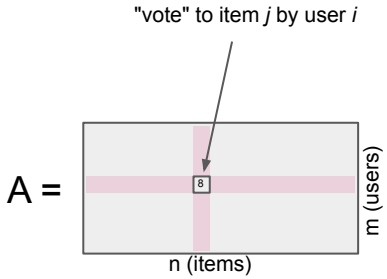


Collaborative Filtering

Matrix-factorization-based recommendation system

Users "collaborate" in identifying your tastes

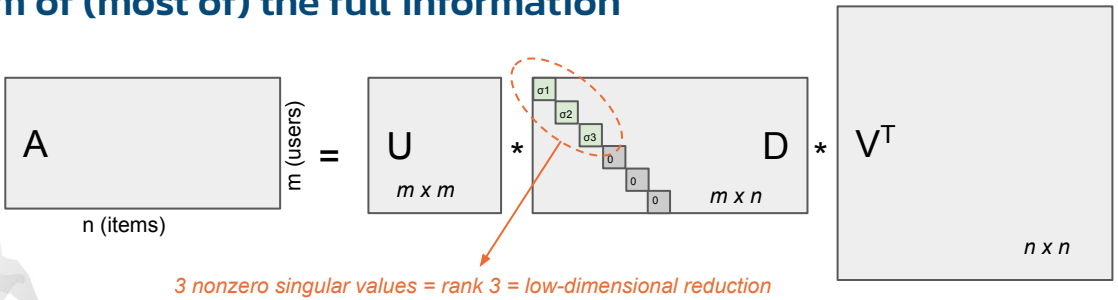
Idea: "If you share the wine and snack tastes of a group of users, you will also like the same meals"



More precisely (model-based filtering):

- arrange preferences in a matrix A (users x items)
- Recast the matrix (*singular value decomposition)*
- Highlight the low-dimensional *latent space*

This finds the "core groups and their core tastes", an optimized form of (most of) the full information



01

Machine Learning
Definitions, Process, Metrics

02

Setup your env
Database, IDE, Tools

03

Learn about your tools
Cassandra, Jupyter, Spark

04

Algos 1
Clustering
Inference

05

Algos 2
Classification
Recommendation

06

What's next?
Quiz, Homework, Next week

Agenda

Homework

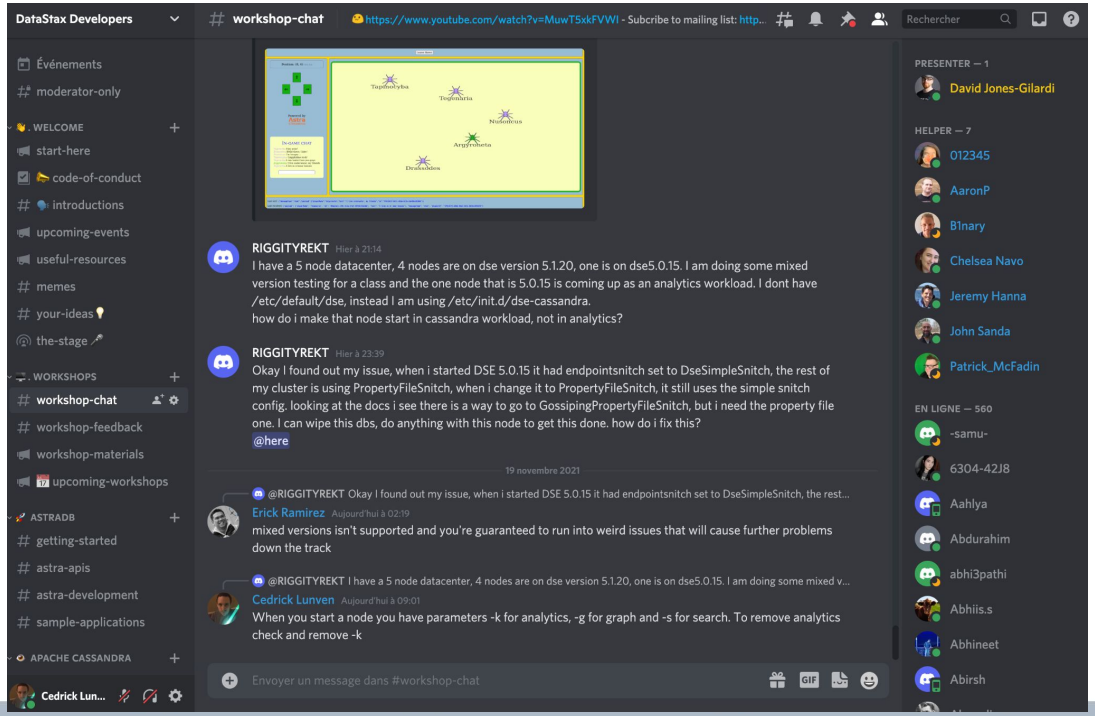
⌨ `!homework`

To gain your Badge:

- **answer some "theory" questions**
- **do the assignment: improve the accuracy of the Random Forest classifier. Details in the README.**

Join our 17k Discord Community

DataStax Developers



!discord

dtsx.io/discord

Discord Community

to NoSQL

ys •
ed 1 week ago

Crash Course | Introduction to Cassandra for Developers

331 views • 1 week ago

Introduction to NoSQL Databases

3.5K views •
Streamed 1 week ago

Introduction to NoSQL Databases

10K views •
Streamed 1 week ago

Build your own NETFLIX clone!

4K views •
Streamed 2 weeks ago

Build your own NETFLIX clone!

7.4K views •
Streamed 2 weeks ago

Astra Streaming Demo

177 views • 2 weeks ago

Kubernetes Ingress Management with Traefik...

496 views •
Streamed 2 weeks ago

Build your own TikTok clone!

1.9K views •
Streamed 3 weeks ago

Build your own TikTok Clone!

4K views •
Streamed 3 weeks ago

How to use the Connect Driver in Astra DB

113 views • 4 weeks ago

How to use the CQL Console in Astra DB

39 views • 4 weeks ago

How to create an Authentication Token in...

37 views • 4 weeks ago

How to use the Data Loader in Astra DB

62 views • 4 weeks ago

Astra DB Sample App Gallery

36 views • 4 weeks ago

How to use Secure Connect in Astra DB

42 views • 4 weeks ago

Cassandra Day India: CL Room (Workshops)

2.4K views •
Streamed 4 weeks ago

Cassandra Da
Room (Talk

1.3K view
Stream

Thank You!