DATASTAX®

Introduction to Machine Learning

with Apache Cassandra™ and Apache Spark™

Aleks Volochnev | 23.01.2020 | Webinar

Hi, I’m Aleks! Welcome to the Webinar!

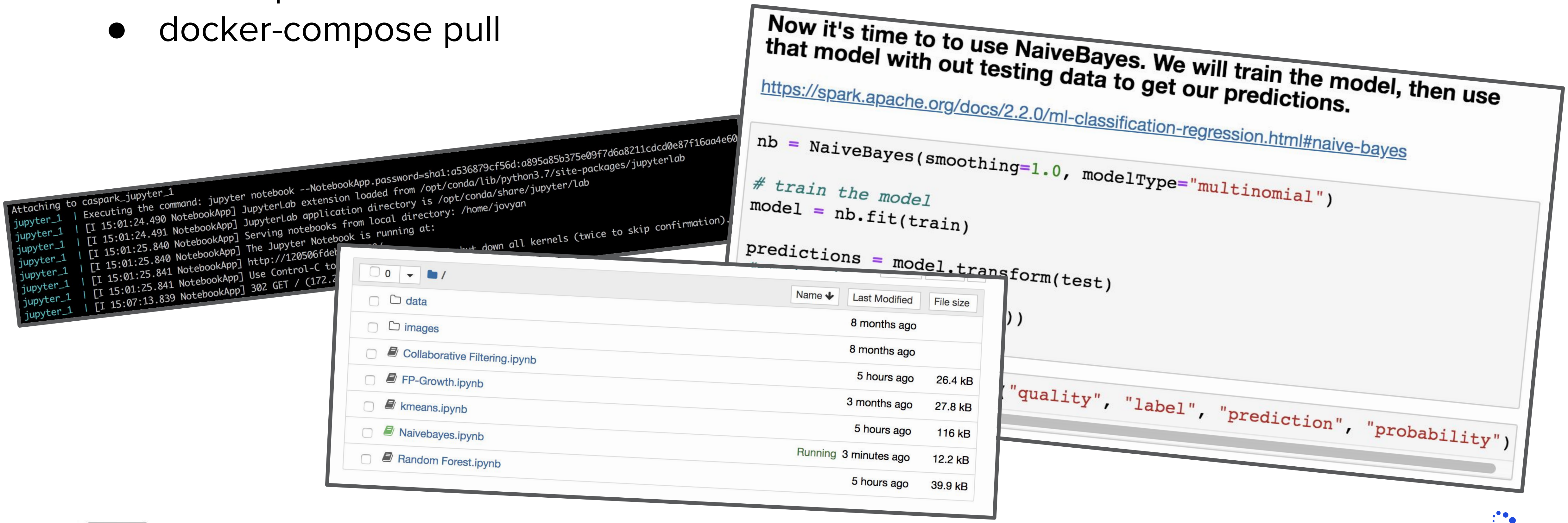
Aleks Volochnev
Developer Advocate at DataStax
@HadesArchitect

After many years in software development as a developer, technical lead, devops engineer and architect, Aleks focused himself on distributed applications and cloud architecture. Working as a developer advocate at DataStax, he shares his knowledge and expertise in the field of microservices, disaster tolerant systems and hybrid platforms.

Before You Start [optional]

To be able to do exercises, you have to configure the environment. It’s optional but recommended.

- git clone https://github.com/HadesArchitect/CaSpark.git
- cd CaSpark
- docker-compose pull



```
Attaching to caspark_jupyter_1
jupyter_1  | Executing the command: jupyter notebook --NotebookApp.password=sha1:a536879cf56d:a895a85b375e09f7d6a8211cdcd0e87f16aa4e60
jupyter_1  | [I 15:01:24.490 NotebookApp] JupyterLab extension loaded from /opt/conda/lib/python3.7/site-packages/jupyterlab
jupyter_1  | [I 15:01:24.491 NotebookApp] JupyterLab application directory is /opt/conda/share/jupyter/lab
jupyter_1  | [I 15:01:25.840 NotebookApp] Serving notebooks from local directory: /home/jovyan
jupyter_1  | [I 15:01:25.840 NotebookApp] The Jupyter Notebook is running at:
jupyter_1  | [I 15:01:25.841 NotebookApp] http://120506fdeb
jupyter_1  | [I 15:01:25.841 NotebookApp] Use Control-C to                    ut down all kernels (twice to skip confirmation).
jupyter_1  | [I 15:07:13.839 NotebookApp] 302 GET / (172.2
```

Name ↓	Last Modified	File size
□ data	8 months ago	
□ images	8 months ago	
□ Collaborative Filtering.ipynb	5 hours ago	26.4 kB
□ FP-Growth.ipynb	3 months ago	27.8 kB
□ kmeans.ipynb	5 hours ago	116 kB
□ Naivebayes.ipynb	Running 3 minutes ago	12.2 kB
□ Random Forest.ipynb	5 hours ago	39.9 kB

Now it's time to to use NaiveBayes. We will train the model, then use that model with out testing data to get our predictions.

https://spark.apache.org/docs/2.2.0/ml-classification-regression.html#naive-bayes

```
nb = NaiveBayes(smoothing=1.0, modelType="multinomial")

# train the model
model = nb.fit(train)

predictions = model.transform(test)

                                    ))

("quality", "label", "prediction", "probability")
```

bit.ly/caspark-webinar

Machine Learning

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

What is 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

What is Machine Learning?

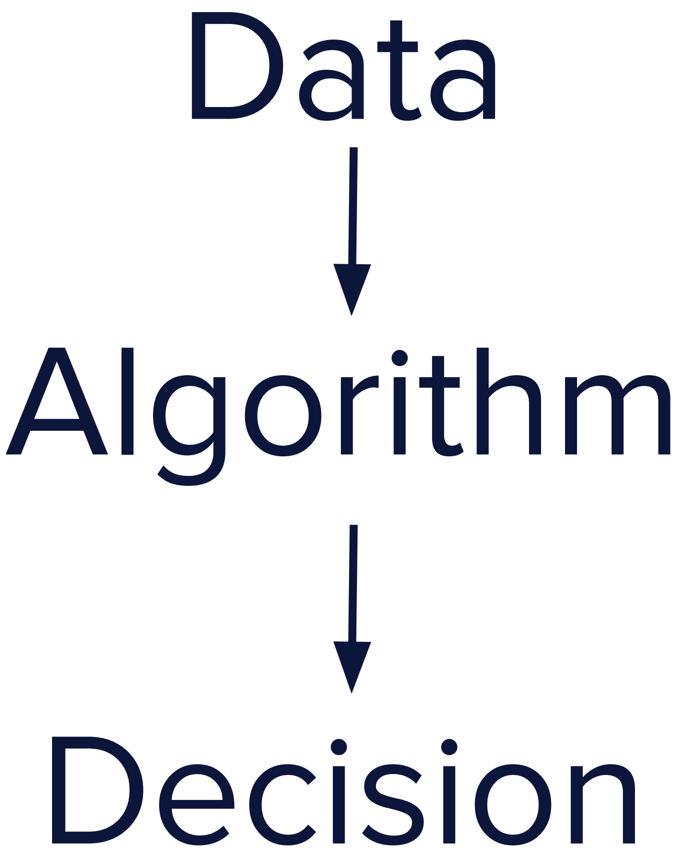
> “Machine Learning is a science of drawing circles [and colorizing them]”
>
> *A. Volochnev*

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

Use Cases

- Forecasts
  - Price
  - Rating
  - Weather
- Aberration Detection
  - Fraud
  - Intrusion
  - Disease
- Classification
  - Face Recognition
  - Categorisation
  - Spam Detection
- Recommendation
- Navigation
- And many others...



Data

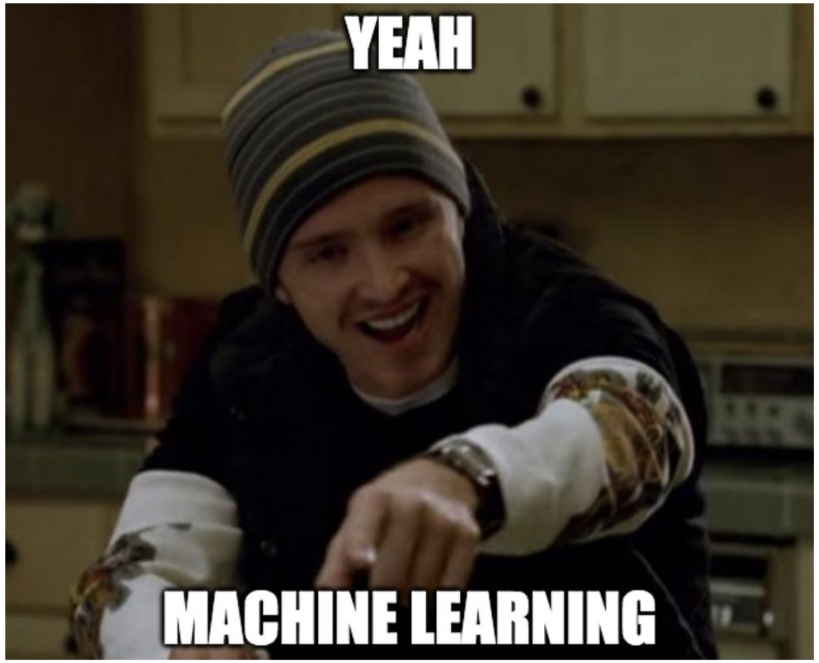
Algorithm

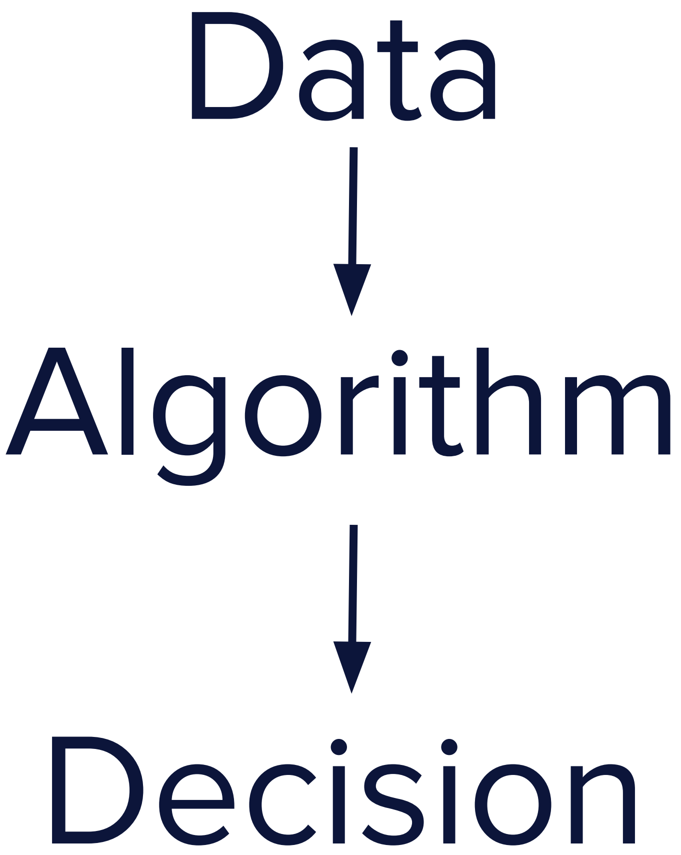
Decision

How it works?

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!





Data

Algorithm

Decision

Algorithms

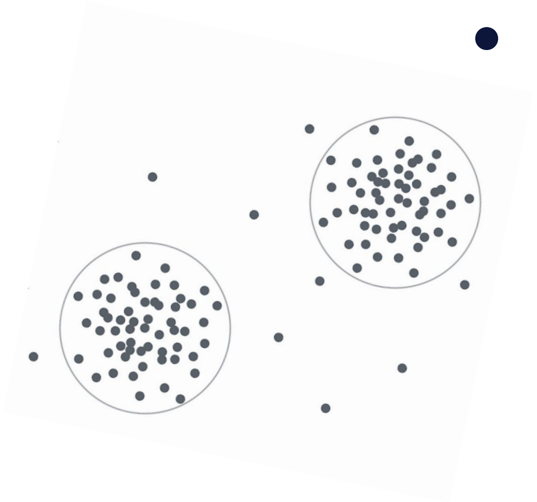
- FP-Growth
- K-Means Clustering
- Naive Bayes
- Decision Trees
- Neural Networks
- Collaborative Filtering
- Logistic Regression
- Support Vector Machines
- Linear Regression
- Apriori Algorithm
- Case-based Reasoning
- Dimensionality Reduction Algorithms
- Gradient Boosting Algorithms
- Hidden Markov Models
- Self-organizing Map
- K-Nearest Neighbour
- ECLAT

And still counting...

Supervised vs Unsupervised

- **Data is labeled or must be labeled**
  - This email is a spam
  - This operation is a fraud
  - This subscription is cancelled
  - etc.
- **Classification is a supervised method**
- **Regression is a supervised method**
- **Easy to test**

- **Data is not labeled**
- **Helps to finds unknown patterns in data**
- **Used for Clustering**
- **Anomaly Detection**
- **Associations**
- **Very useful in exploratory analysis**
- **Allows Dimensionality Reduction**
- **Hard to test**

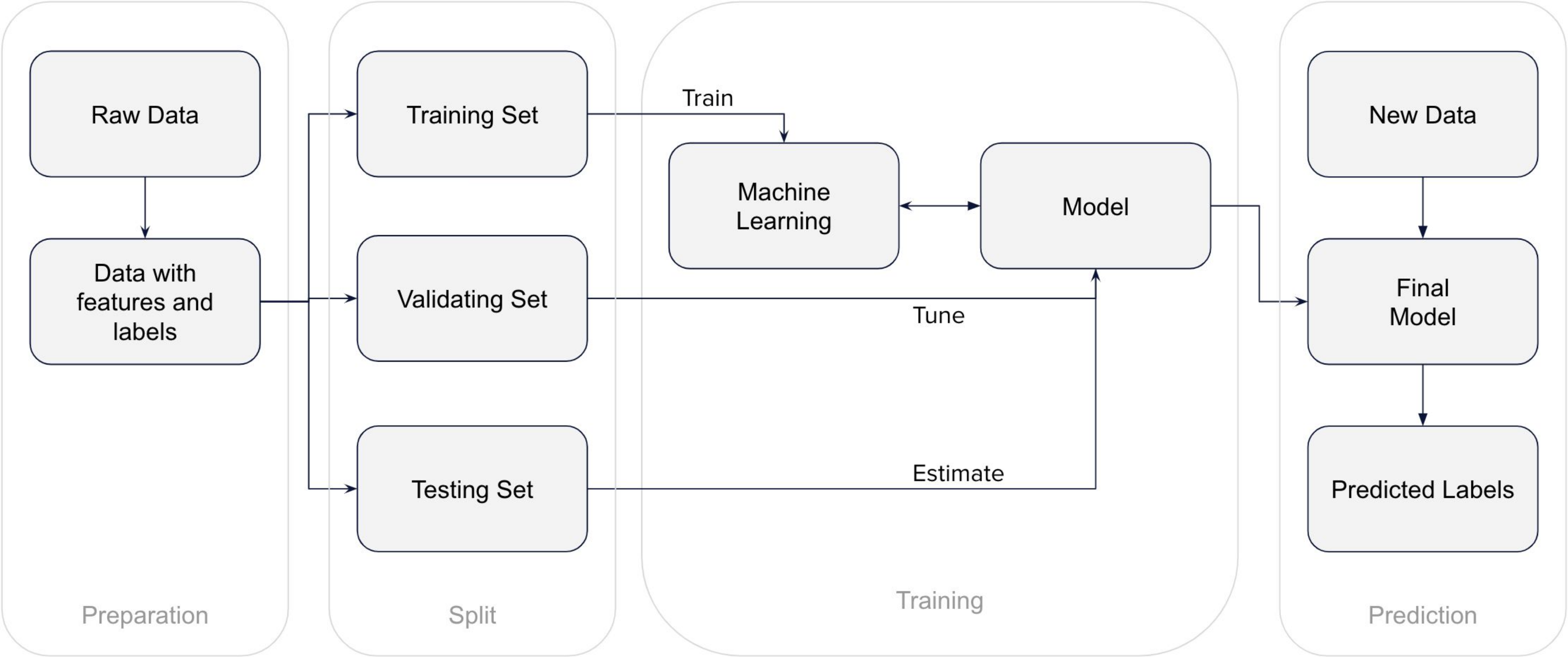


Learning Workflow

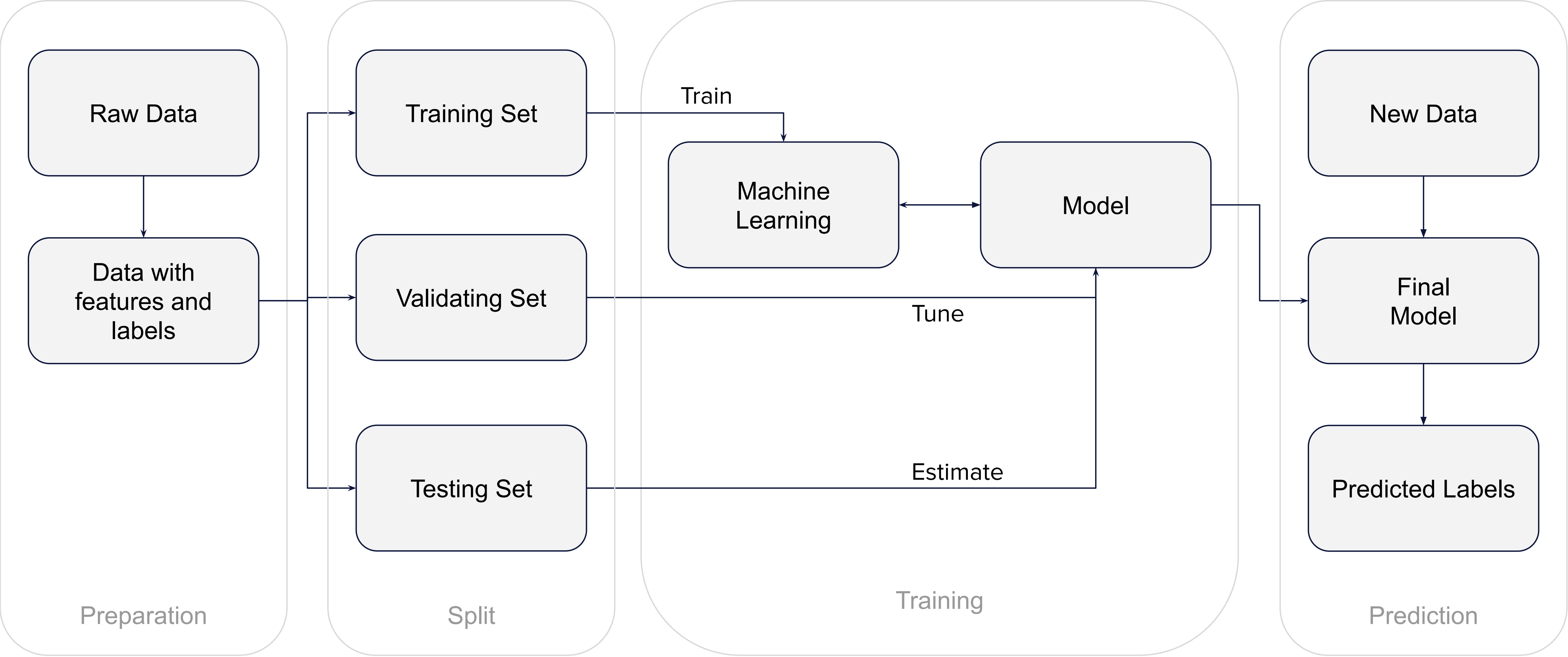
Step by Step

Learning Workflow

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



Learning Workflow



Raw Data

Data with features and labels

Preparation

Training Set

Validating Set

Testing Set

Split

Train

Machine Learning

Model

Tune

Estimate

Training

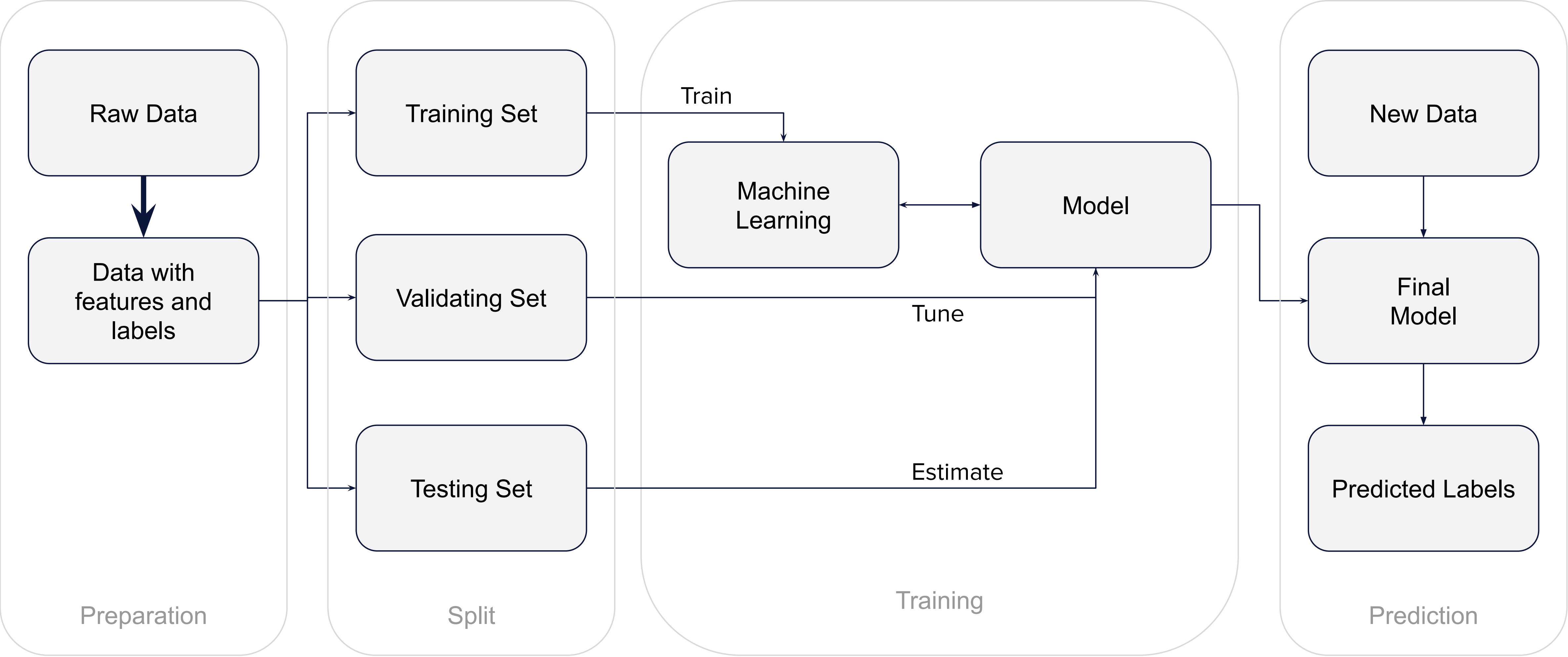
New Data

Final Model

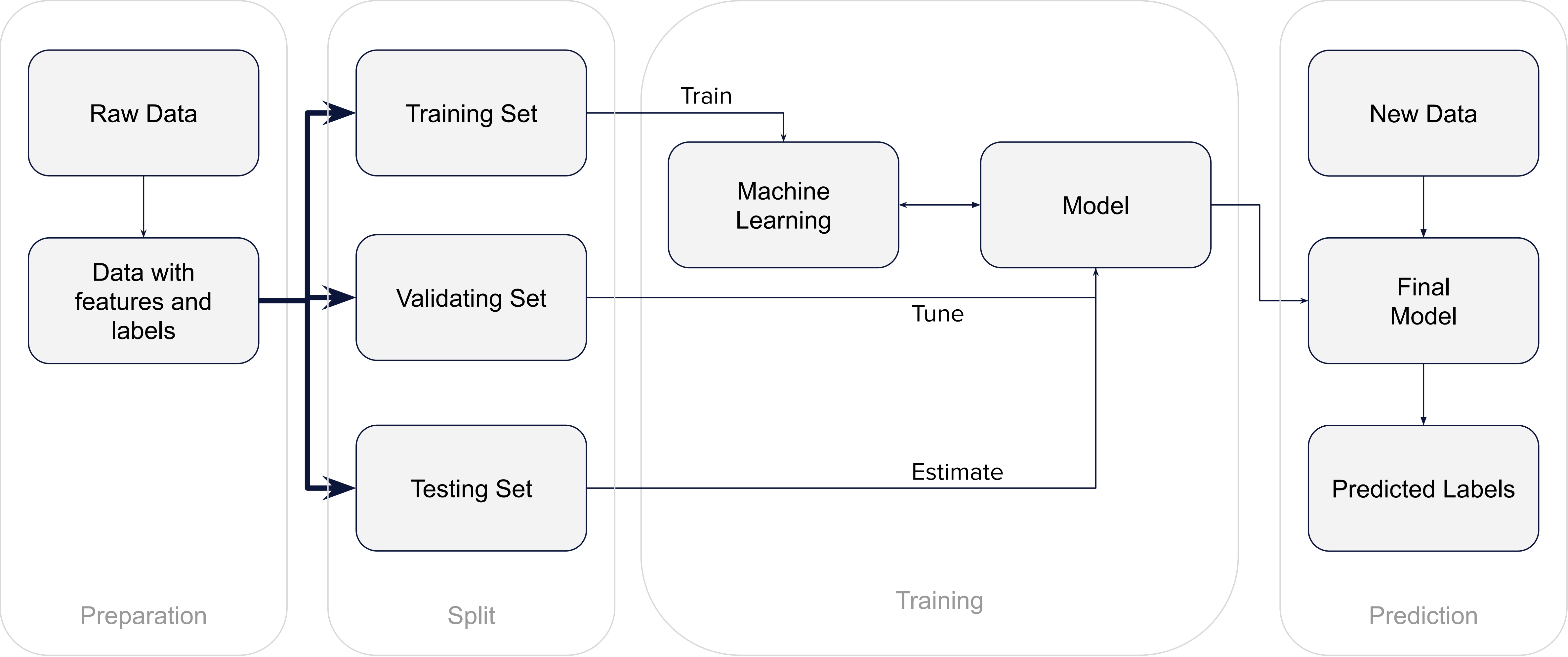
Predicted Labels

Prediction

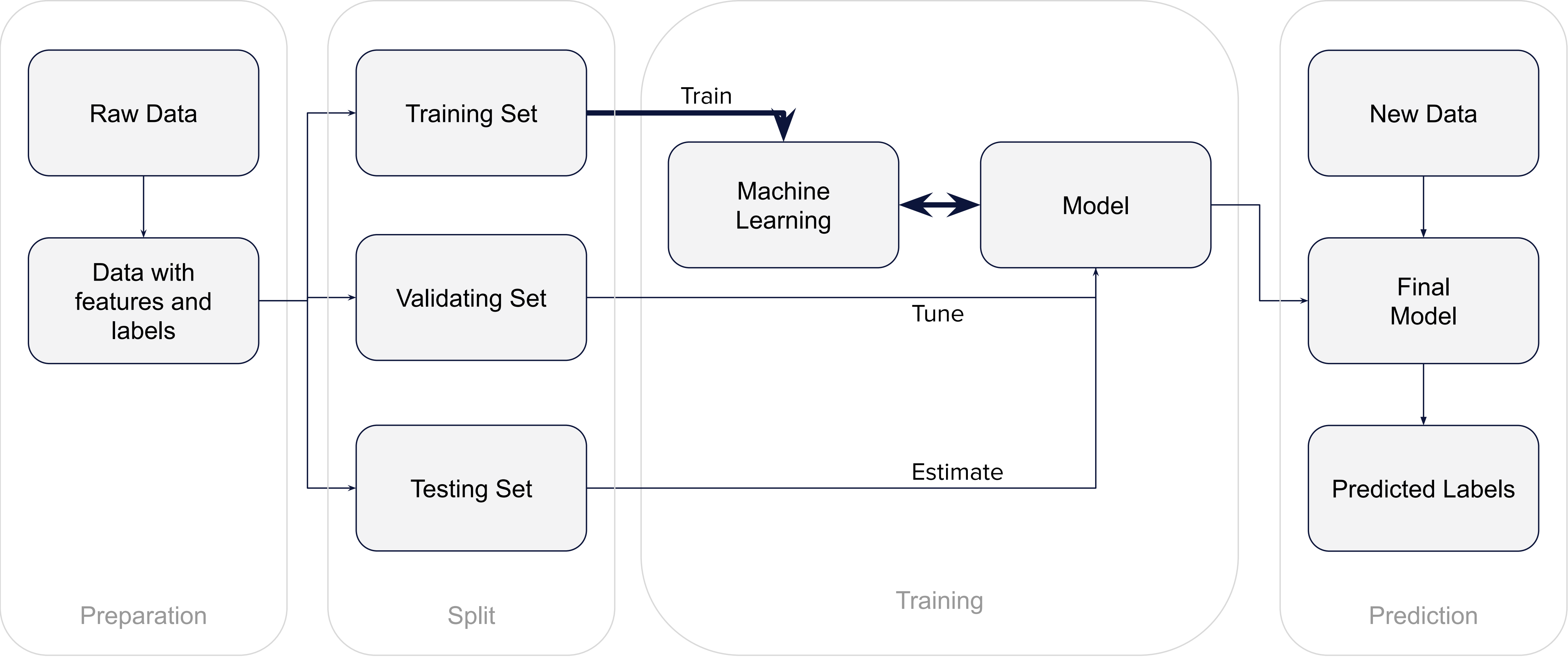
Learning Workflow: Preparation



Learning Workflow: Split



Learning Workflow: Training



Raw Data

Data with features and labels

Preparation

Training Set

Validating Set

Testing Set

Split

Train

Machine Learning

Model

Tune

Estimate

Training

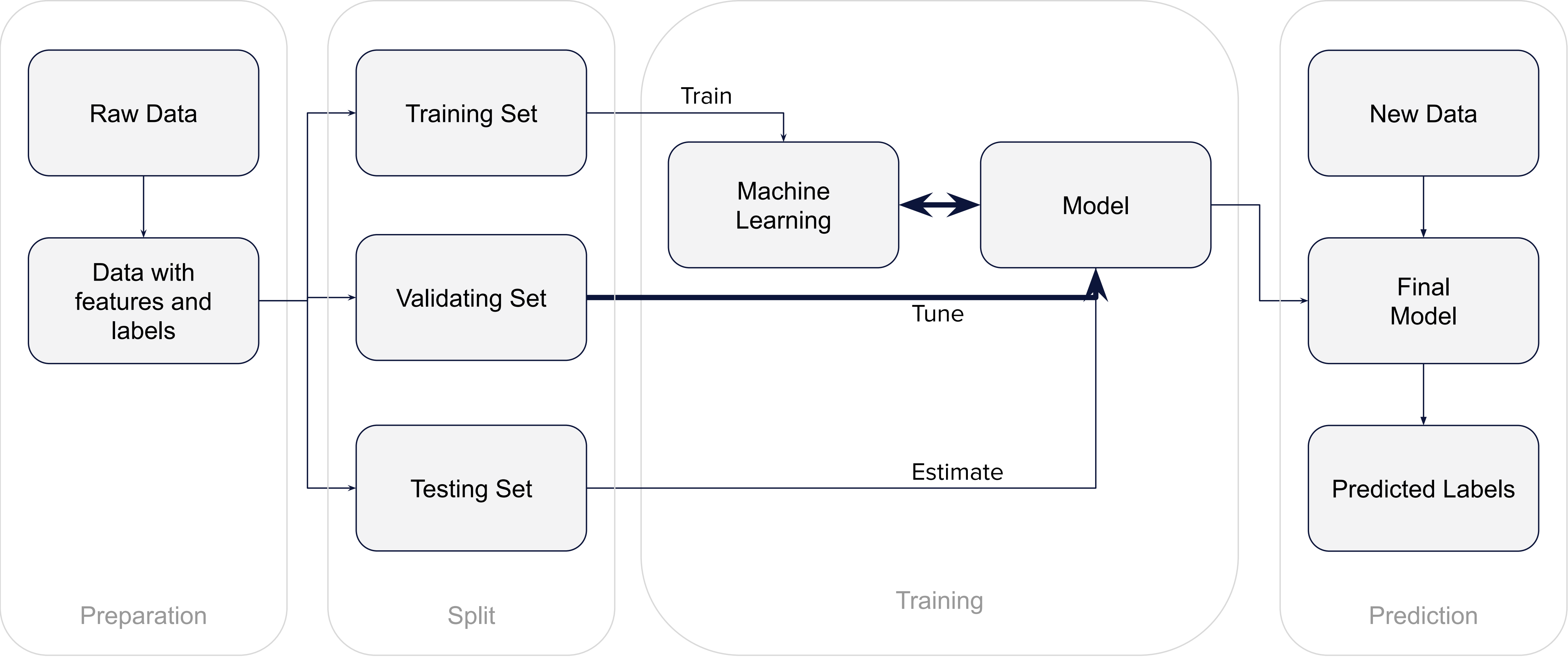
New Data

Final Model

Predicted Labels

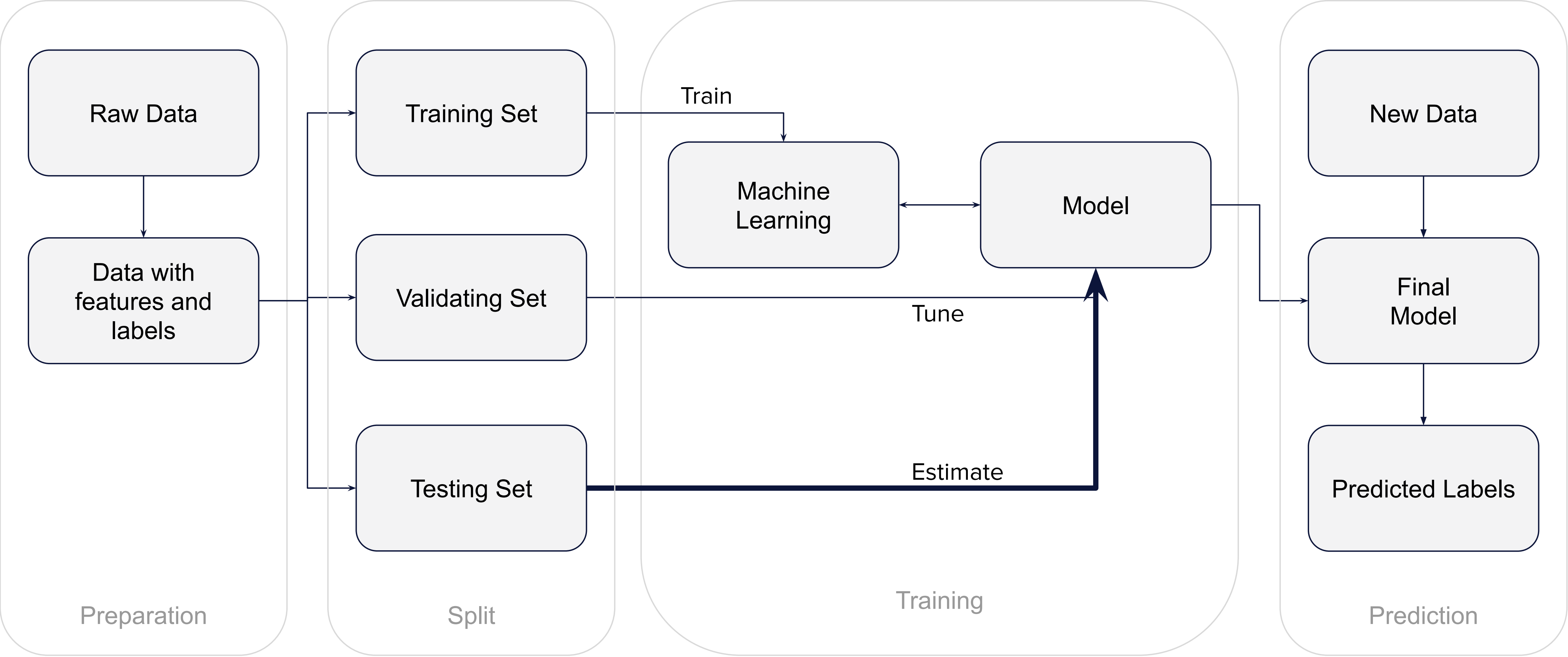
Prediction

Learning Workflow: Tuning

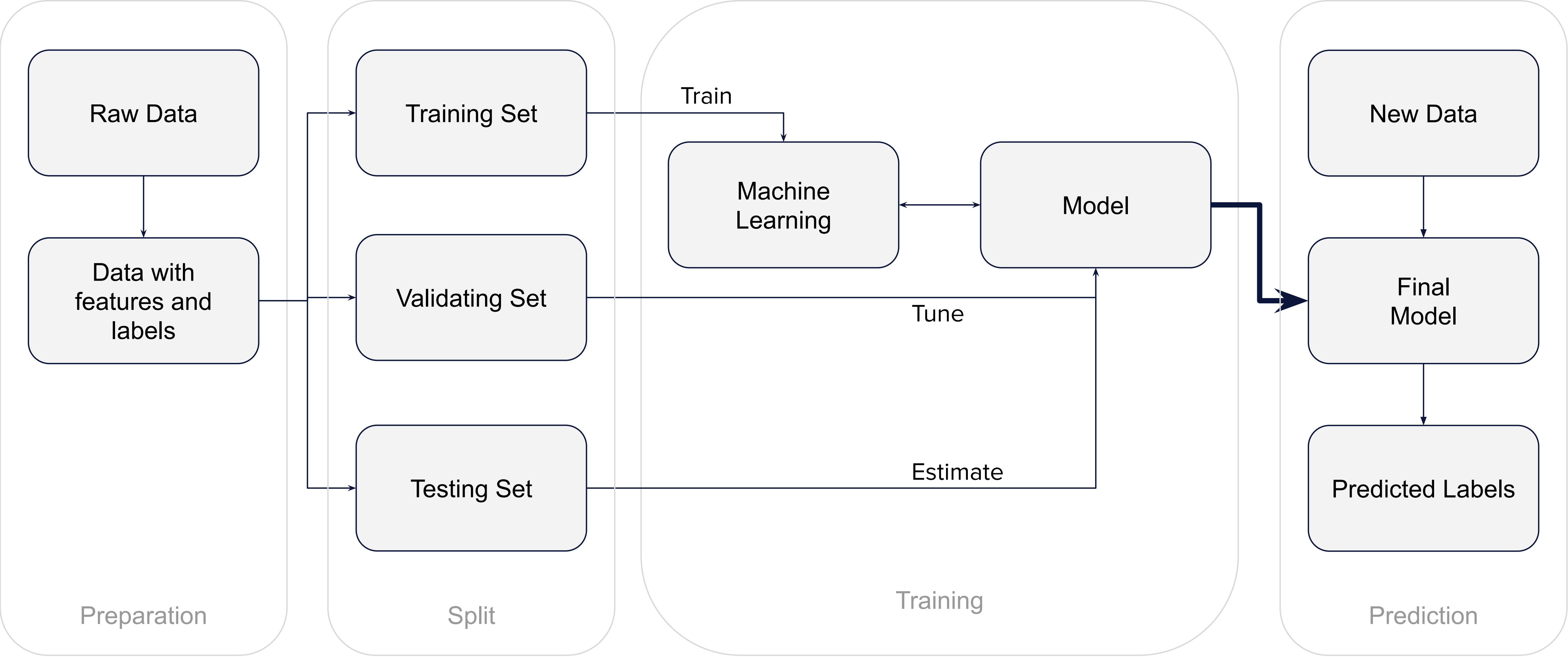


DATASTAX

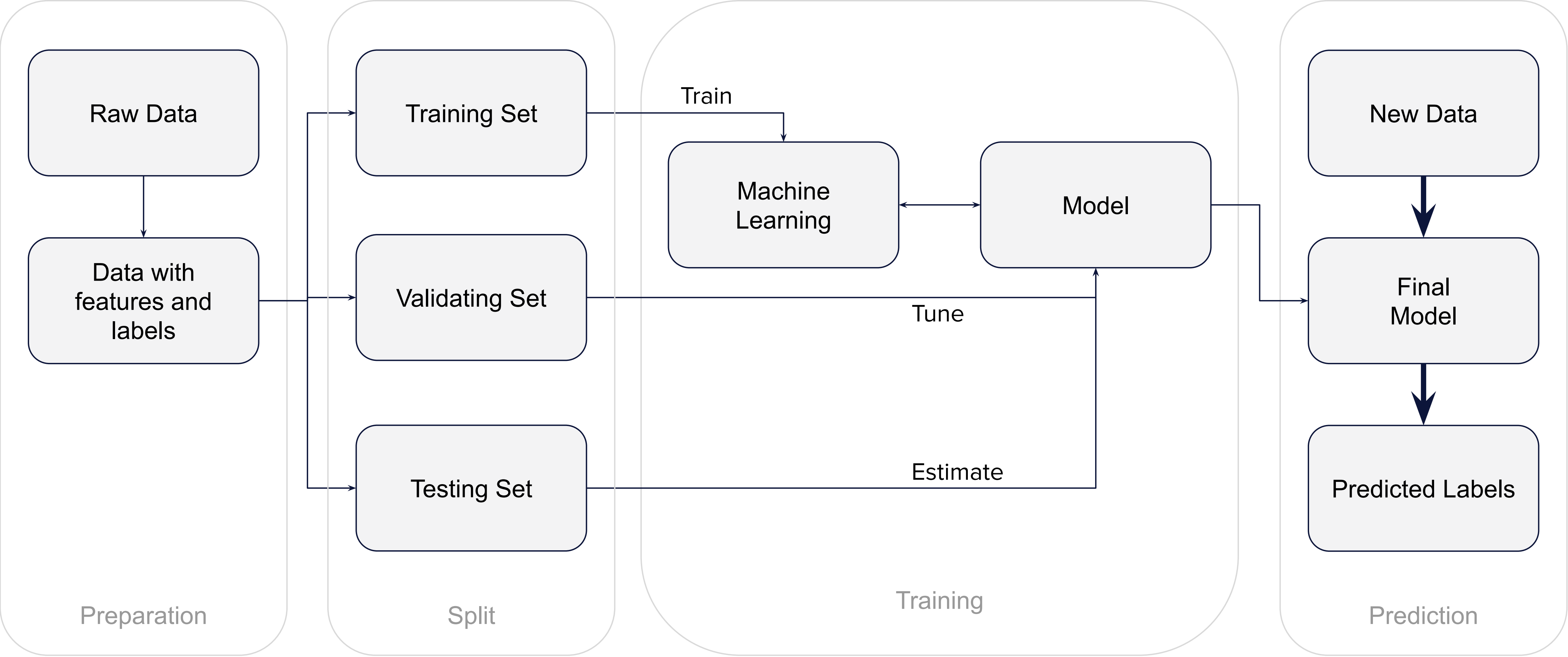
Learning Workflow: Testing



Learning Workflow: Final Model



Learning Workflow: Prediction



Metrics

You can’t control that you can’t measure

Accuracy

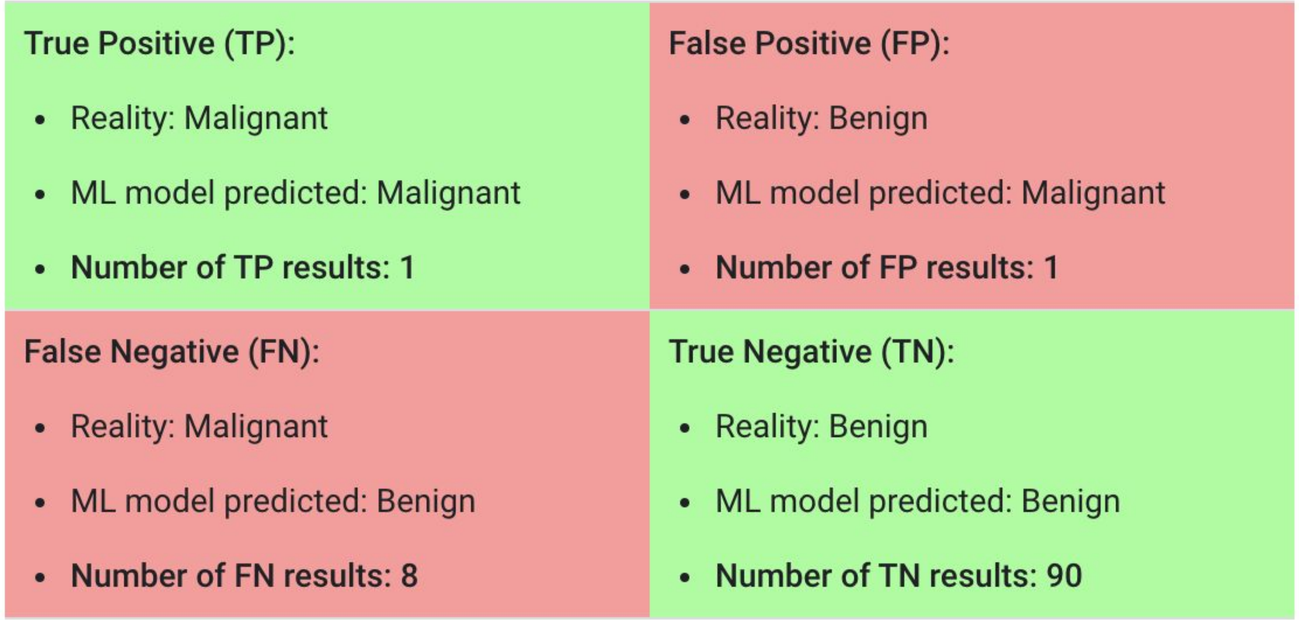
Accuracy is an evaluating classification models metric, it is the fraction of predictions model identified correctly.

Accuracy = Correct Predictions / Total Predictions

In the example, we have accuracy 91/100 = 0.91
Pretty high! The model looks good, isn't it?

Tumor Example

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

Accuracy

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

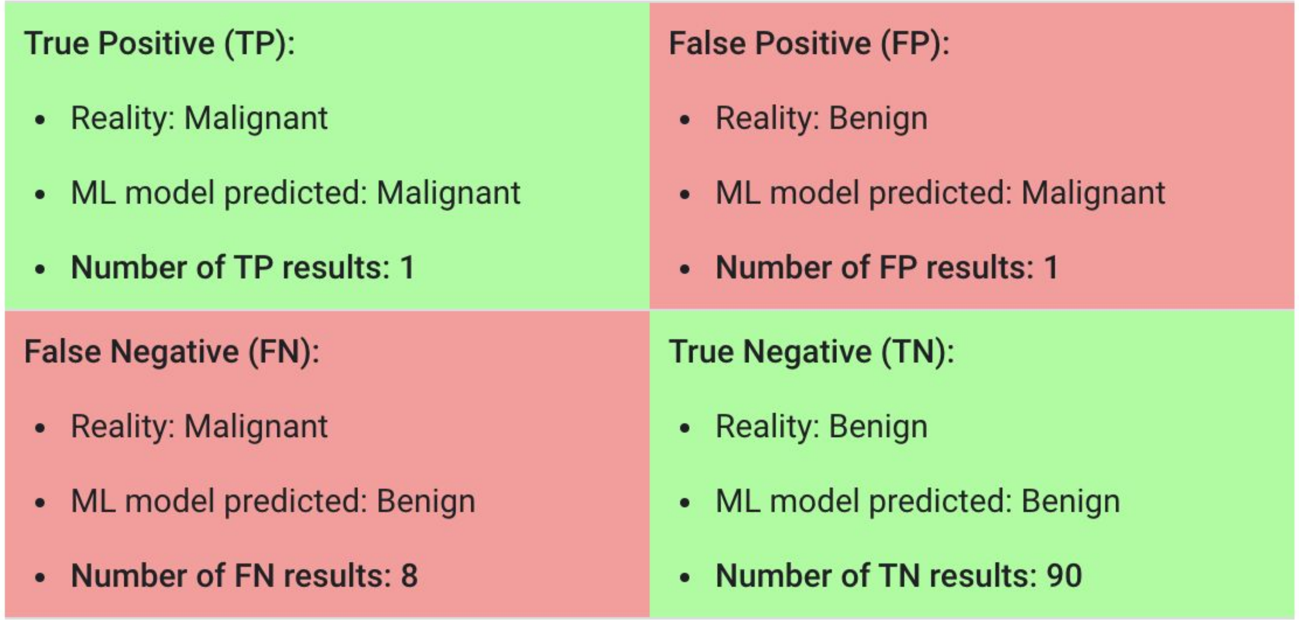
Accuracy = Correct Predictions / Total Predictions

In the example, we have accuracy 91/100 = 0.91
Pretty high! The model looks good, isn’t it?

Well, we just sent back home 8 people without proper treatment! *We could have better result by literally* ***throwing a coin!***

Tumor Example

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

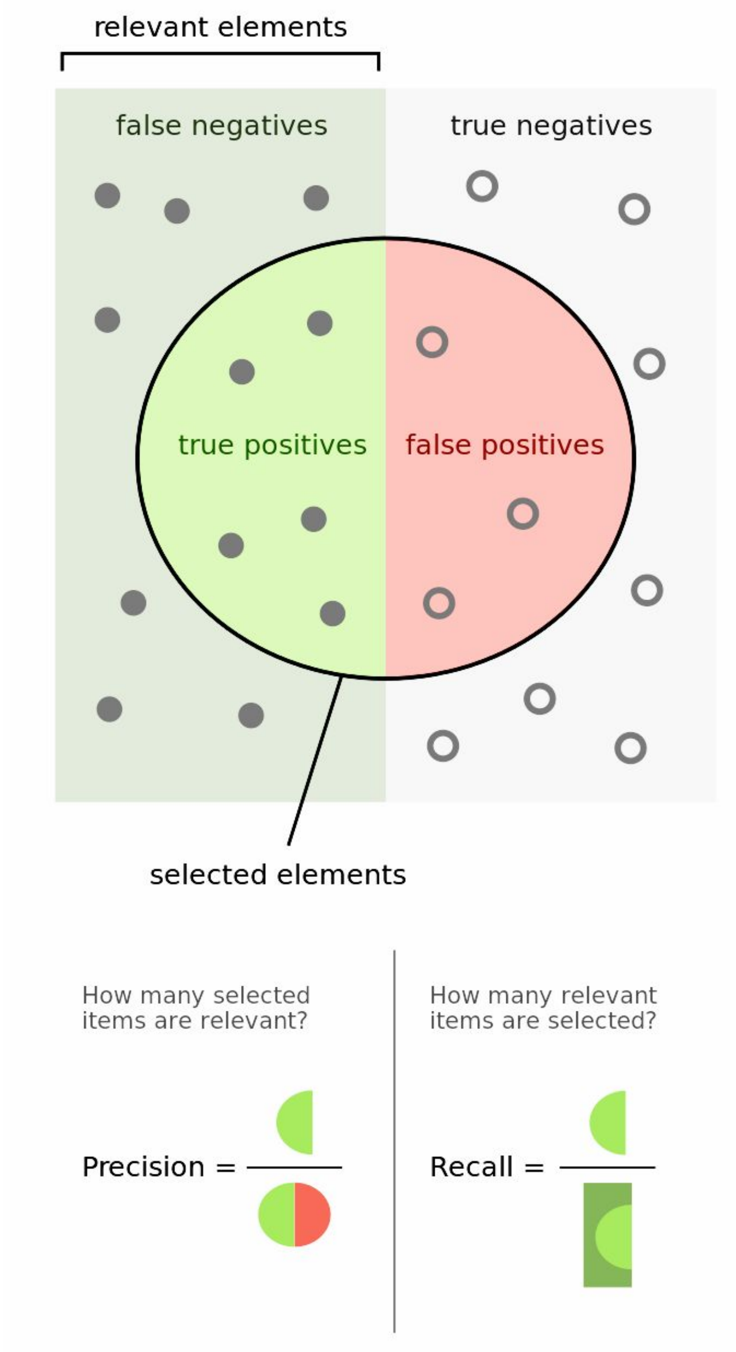
Precision vs Recall

Positive Predictive Value

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

Precision =
True positives / All positives

For the Tumor example,
Precision is 1 / (1 + 1) = **0.5**



Sensitivity

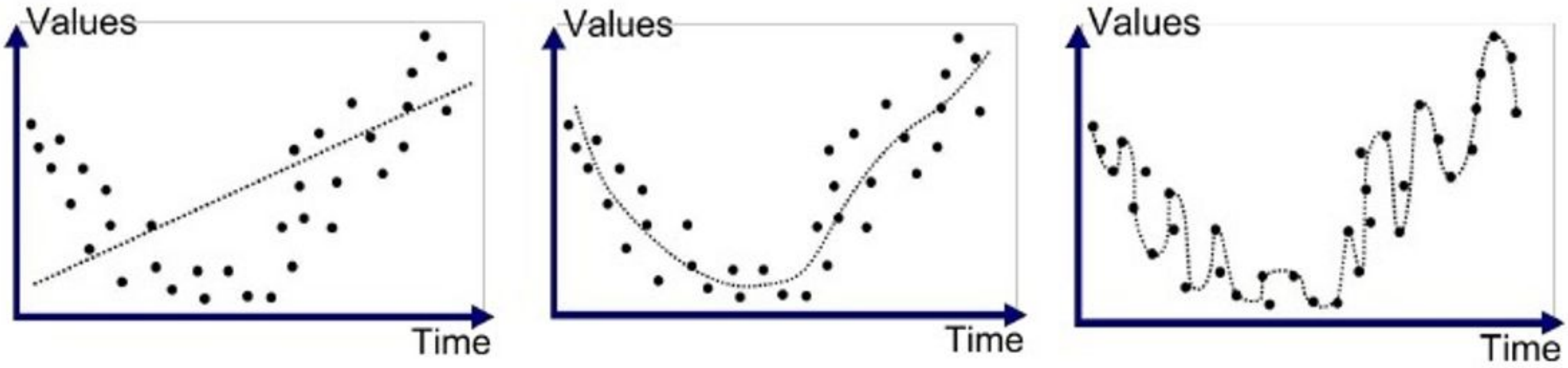
Recall counts correctly identified positives out of all real positives.

Recall =
True Positives / All Real Positives

For the Tumor example,
Recall is 1 / (1 + 8) = **0.11**

Under-fitted vs Over-fitted Model

Anub Bhande @ Medium.com



Underfitted

Not accurate, too simple

Good Fit/Robust

Good, well generalised

Overfitted

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

Speech is silver, PRACTICE is golden

Start your Engines

Know your Tools

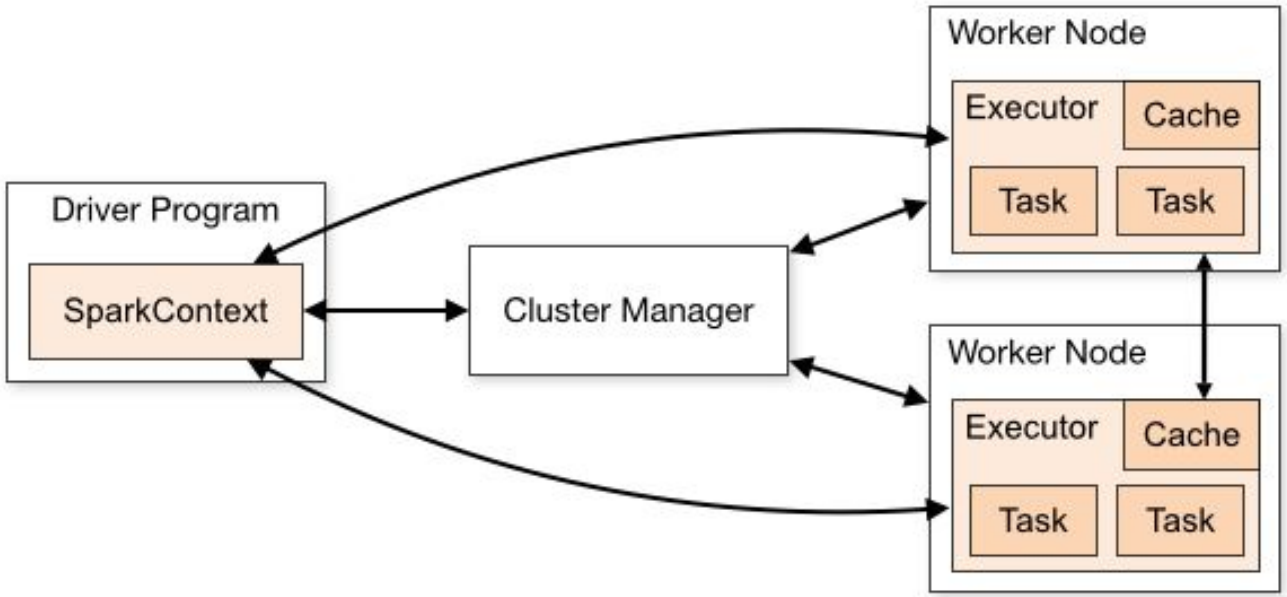
- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- Python
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

SURVIVE THE ZOMBIE APOCALYPSE
KNOW YOUR TOOLS
CRICKET BAT
ZOMBIES HAVE SOFT SKULLS. TAKE ADVANTAGE.
AXE
LIKE A CRICKET BAT. BUT BETTER.
CHAINSAW
USEFUL FOR SPEEDY LIMB REMOVAL AND DECAPITATIONS.
SHOTGUN
NOW WE'RE TALKING. AIM FOR THE FACE.
GATLING GUN
AHHH YEAH JUST GO NUTS.
WHEN ALL ELSE FAILS
JUST GRAB ANYTHING AND FIGHT FOR YOUR LIFE!

Know your Tools

- **Apache Spark**
- Apache Cassandra
- Jupyter
- DS Studio
- Python
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

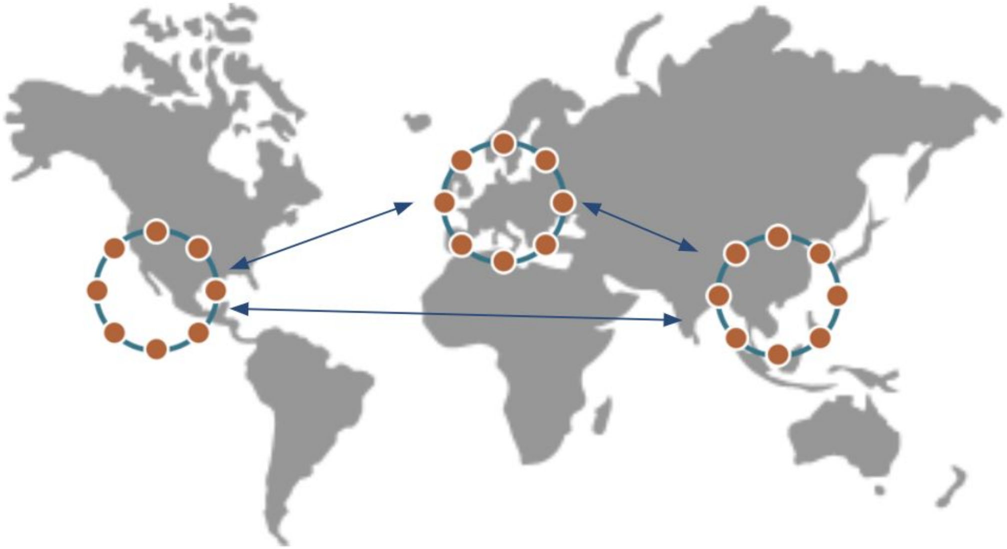
Apache Spark is an open-source distributed general-purpose cluster-computing framework. Spark provides an interface for programming entire clusters with implicit data parallelism and fault tolerance.



Know your Tools

- Apache Spark
- **Apache Cassandra**
- Jupyter
- DS Studio
- Python
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

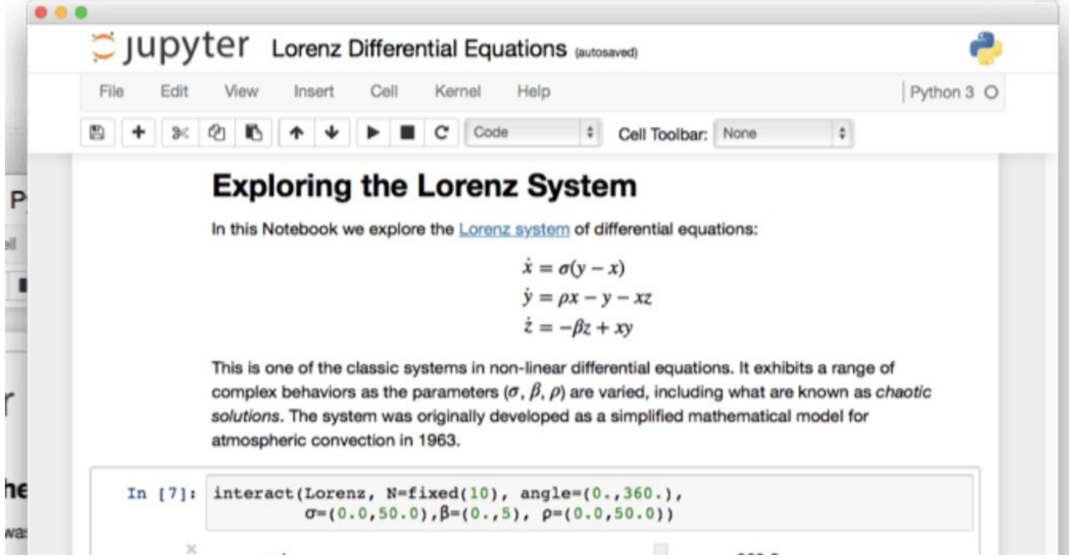
Apache Cassandra is a free open-source distributed, decentralized NoSQL database designed to handle huge amounts of data across multiple servers and data centers, providing highest availability and performance.



Know your Tools

- Apache Spark
- Apache Cassandra
- **Jupyter**
- DS Studio
- Python
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

The Jupyter Notebook is an open-source web application that allows to create and share documents that contain live code, equations, visualizations etc. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.



Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

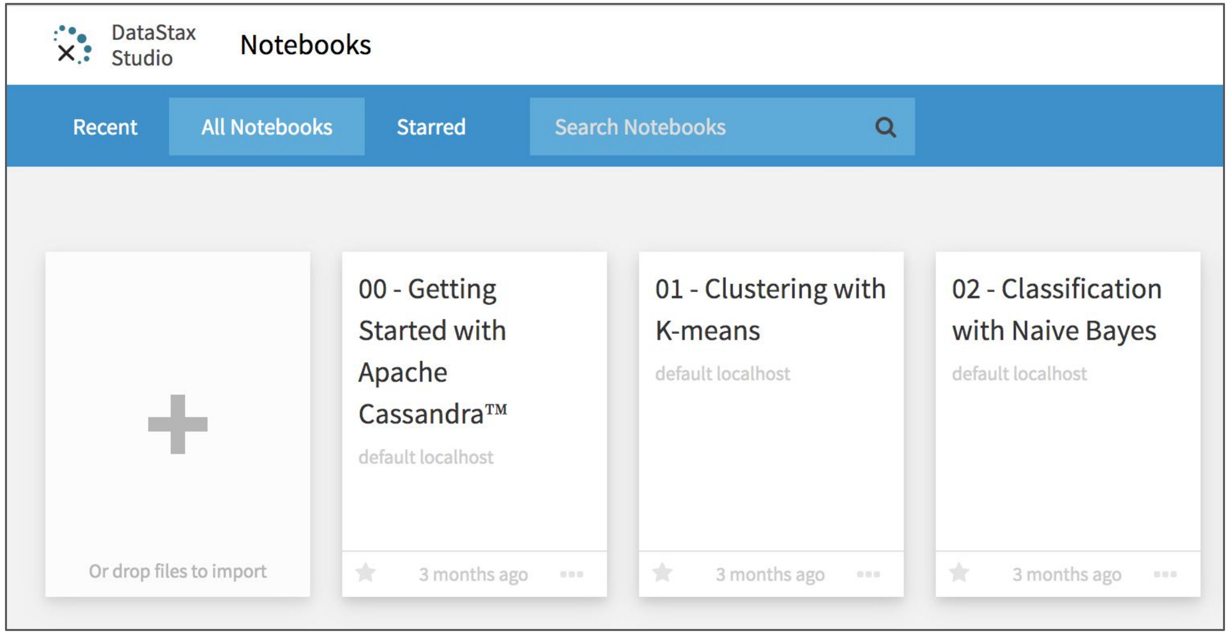
This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

Know your Tools

- Apache Spark
- Apache Cassandra
- Jupyter
- **DS Studio**
- Python
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

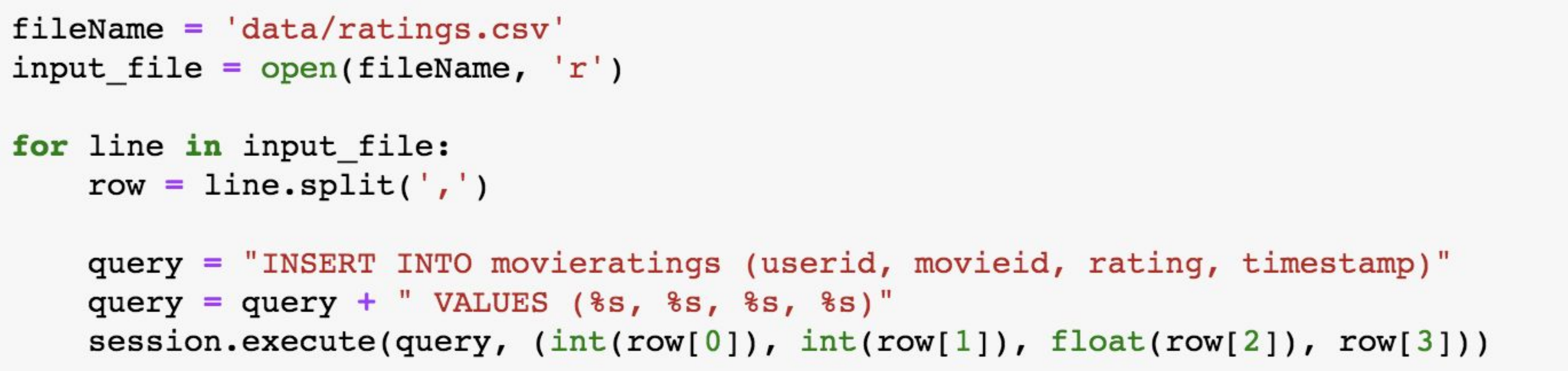
DataStax Studio 6.0 is an interactive developer tool for CQL (Cassandra Query Language), Spark SQL, and DSE Graph. DataStax Studio has all the tools needed for ad hoc queries, visualizing and exploring data sets, profiling performance and comes with a notebook interface that fuels collaboration.



Know your Tools

- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- **Python**
  - Pandas
  - PySpark
  - NumPy
  - Scikit Learn

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.

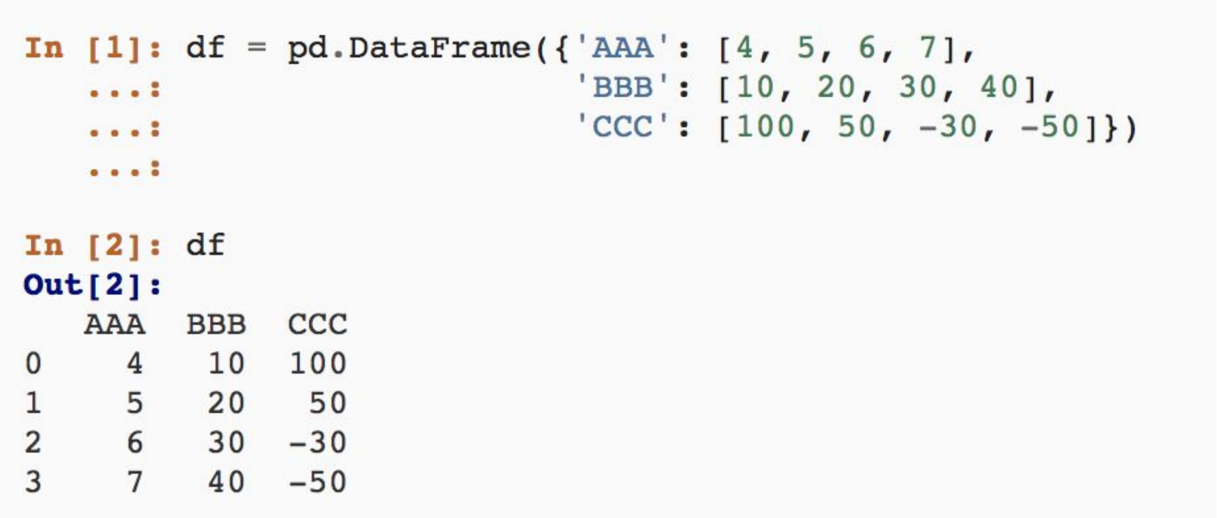


```python
fileName = 'data/ratings.csv'
input_file = open(fileName, 'r')

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]))
```

Know your Tools

- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- Python
    - **Pandas**
    - PySpark
    - NumPy
    - Scikit Learn

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.



```
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
```

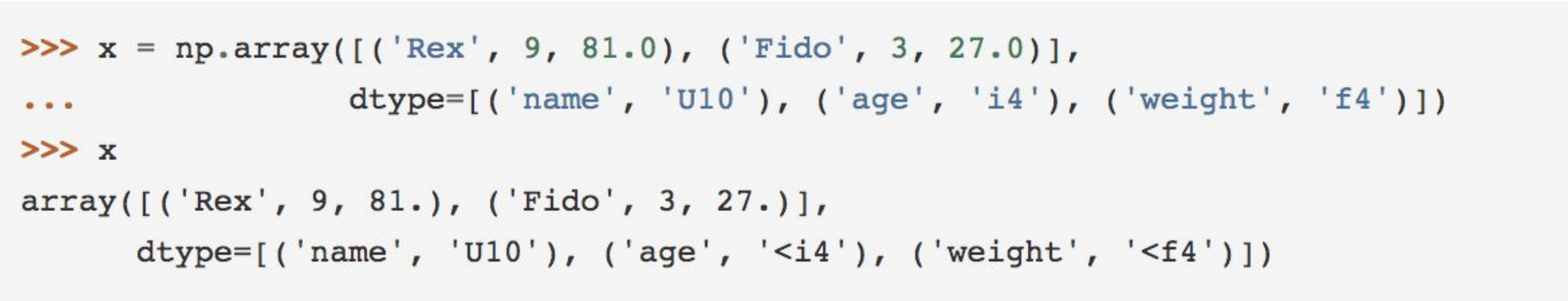
Know your Tools

- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- Python
    - Pandas
    - **PySpark**
    - NumPy
    - Scikit Learn

Know your Tools

- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- Python
  - Pandas
  - PySpark
  - **NumPy**
  - Scikit Learn

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.

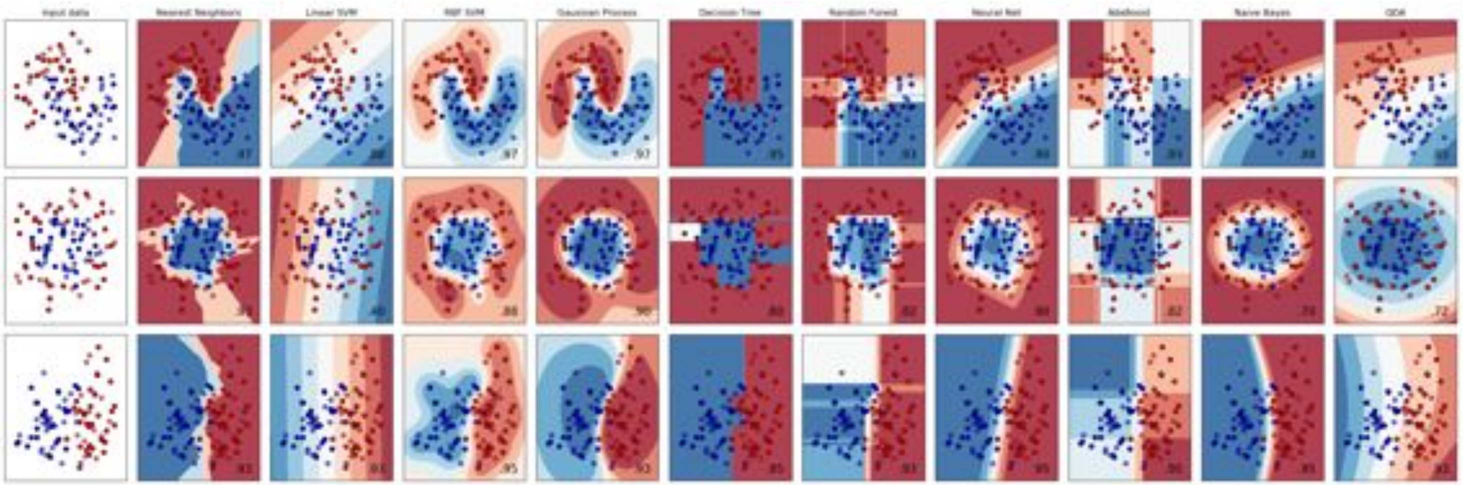


```
>>> x = np.array([('Rex', 9, 81.0), ('Fido', 3, 27.0)],
...              dtype=[('name', 'U10'), ('age', 'i4'), ('weight', 'f4')])
>>> x
array([('Rex', 9, 81.), ('Fido', 3, 27.)],
      dtype=[('name', 'U10'), ('age', '<i4'), ('weight', '<f4')])
```

Know your Tools

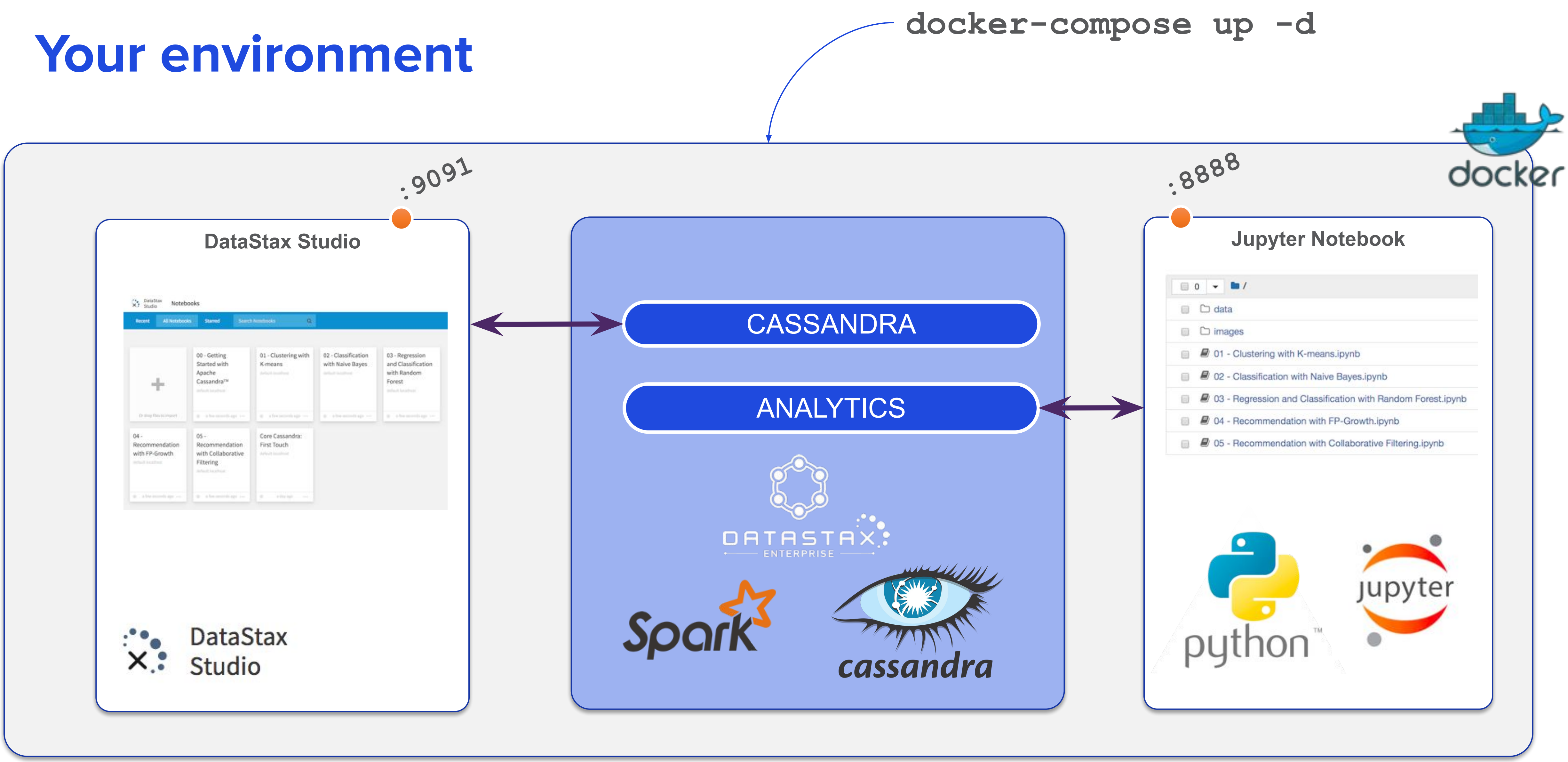
- Apache Spark
- Apache Cassandra
- Jupyter
- DS Studio
- Python
  - Pandas
  - PySpark
  - NumPy
  - **Scikit Learn**

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.



Your environment

```
docker-compose up -d
```



DataStax Studio Notebooks

Recent | All Notebooks | Starred | Search Notebooks

+ Or drop files to import

00 - Getting Started with Apache Cassandra™ default localhost a few seconds ago

01 - Clustering with K-means default localhost a few seconds ago

02 - Classification with Naive Bayes default localhost a few seconds ago

03 - Regression and Classification with Random Forest default localhost a few seconds ago

04 - Recommendation with FP-Growth default localhost a few seconds ago

05 - Recommendation with Collaborative Filtering default localhost a few seconds ago

Core Cassandra: First Touch default localhost a day ago

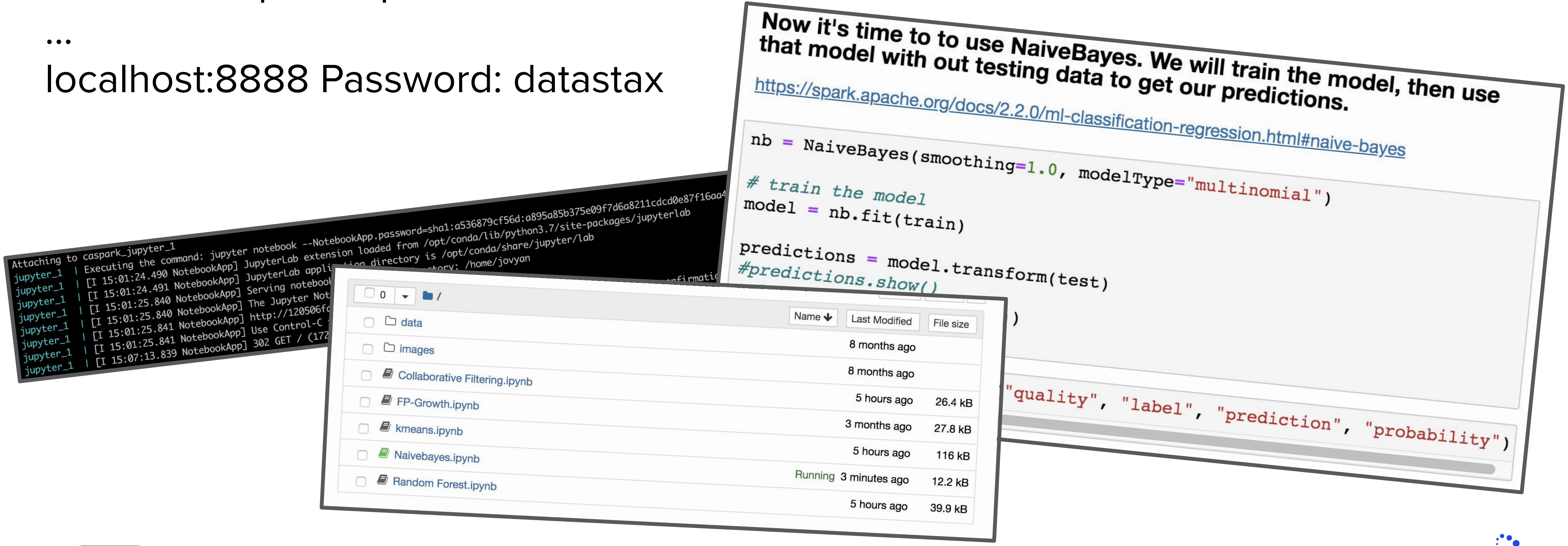
□ 0 ▾ /

- □ data
- □ images
- □ 01 - Clustering with K-means.ipynb
- □ 02 - Classification with Naive Bayes.ipynb
- □ 03 - Regression and Classification with Random Forest.ipynb
- □ 04 - Recommendation with FP-Growth.ipynb
- □ 05 - Recommendation with Collaborative Filtering.ipynb

bit.ly/caspark-webinar

Your environment

git clone https://github.com/HadesArchitect/CaSpark.git
docker-compose up -d
…
localhost:8888 Password: datastax



```
Attaching to caspark_jupyter_1
jupyter_1  | Executing the command: jupyter notebook --NotebookApp.password=sha1:a536879cf56d:a895a85b375e09f7d6a8211cdcd0e87f16aa4
jupyter_1  | [I 15:01:24.490 NotebookApp] JupyterLab extension loaded from /opt/conda/lib/python3.7/site-packages/jupyterlab
jupyter_1  | [I 15:01:24.491 NotebookApp] JupyterLab appli          directory is /opt/conda/share/jupyter/lab
jupyter_1  | [I 15:01:25.840 NotebookApp] Serving noteboo             tory: /home/jovyan
jupyter_1  | [I 15:01:25.840 NotebookApp] The Jupyter Not
jupyter_1  | [I 15:01:25.841 NotebookApp] http://120506fc
jupyter_1  | [I 15:01:25.841 NotebookApp] Use Control-C                                          nfirmatic
jupyter_1  | [I 15:07:13.839 NotebookApp] 302 GET / (172
```

Name	Last Modified	File size
□ data	8 months ago	
□ images	8 months ago	
□ Collaborative Filtering.ipynb	5 hours ago	26.4 kB
□ FP-Growth.ipynb	3 months ago	27.8 kB
□ kmeans.ipynb	5 hours ago	116 kB
□ Naivebayes.ipynb Running	3 minutes ago	12.2 kB
□ Random Forest.ipynb	5 hours ago	39.9 kB

Now it's time to use NaiveBayes. We will train the model, then use that model with out testing data to get our predictions.

https://spark.apache.org/docs/2.2.0/ml-classification-regression.html#naive-bayes

```
nb = NaiveBayes(smoothing=1.0, modelType="multinomial")

# train the model
model = nb.fit(train)

predictions = model.transform(test)
#predictions.show()
                                    )

"quality", "label", "prediction", "probability")
```

bit.ly/caspark-webinar

DATASTAX

DATASTAX

Algorithms in the wild

Start your Engines

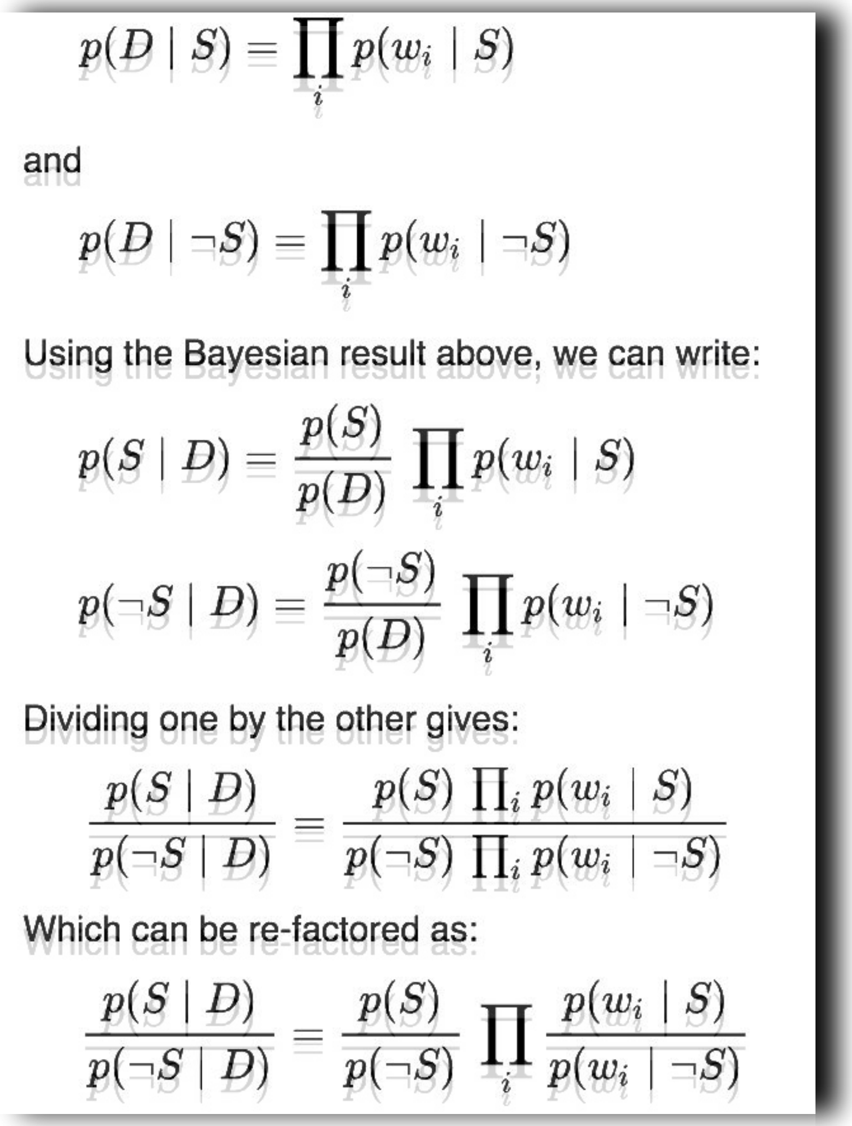
Naïve Bayes

Supervised Classification Algorithm

Naïve Bayes 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:

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

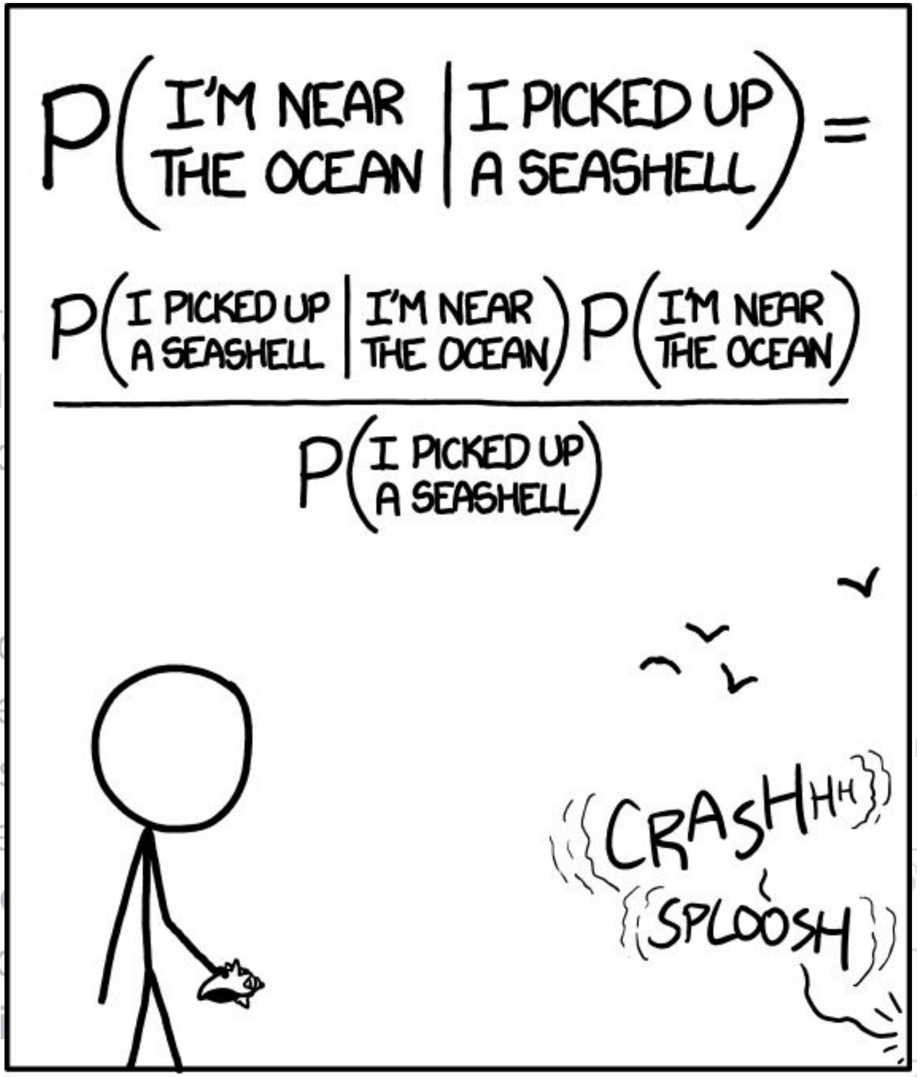
Which can be re-factored as:

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

Naïve Bayes Algo

“Naïve Bayes classifiers are a f
"probabilistic classifiers" based
theorem with strong “naïve” in
between the features.

Naïve Bayes is a popular meth
the problem of judging docume
category or the other (such as s
with word frequencies as the fe
pre-processing, it is competitive
more advanced methods inclu
machines. It also finds applicati
diagnosis.”


P(I'M NEAR THE OCEAN | I PICKED UP A SEASHELL) =
P(I PICKED UP A SEASHELL | I'M NEAR THE OCEAN) P(I'M NEAR THE OCEAN)
P(I PICKED UP A SEASHELL)
CRASHHHH
SPLOOSH

STATISTICALLY SPEAKING, IF YOU PICK UP A SEASHELL AND *DON'T* HOLD IT TO YOUR EAR, YOU CAN PROBABLY HEAR THE OCEAN.

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

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

Bayesian result above, we can write:

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

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

ne by the other gives:

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

n be re-factored as:

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

Naïve Bayes Algorithm

For Humans

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:

- 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

New Email: "Easy Money!" Is it spam?

Human: What is the chance it's spam if it has 'Easy Money'?
Scientist: P(spam | 'Easy', 'Money') = ?

Money:	2 out of 3 spam	=>	P('money' | spam) = ⅔
	1 out of 5 not	=>	P('money' | not) = ⅕
Easy:	1 out of 3 spam	=>	P('money' | spam) = ⅓
	1 out of 5 not	=>	P('money' | not) = ⅕

Conditional Probability! **P(spam) + P(not) = 1.0**

P(spam | 'Easy', 'Money') ∝ P('Easy', 'Money' | spam) * P (spam)

∝ P('Easy'| spam) * P('Money' | spam) * P (spam)

⅓ ⅔ ⅜

P(not | 'Easy', 'Money') ∝ P('Easy', 'Money' | not) * P (not)

∝ P('Easy'| not) * P('Money' | not) * P (not)

⅕ ⅕ ⅝

P(spam | 'Easy', 'Money') ∝ ⅓ * ⅔ * ⅜ ∝ 1 / 12 ∝ 10/13

P(not | 'Easy', 'Money') ∝ ⅕ * ⅕ * ⅝ ∝ 1 / 40 ∝ 3/13

P(spam | 'Easy', 'Money') = 0.7692307 = 76.9%

Naïve Bayes Algorithm

For Humans

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 mostly probably will continue: *"in a galaxy far far away"*. This algorithm 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:

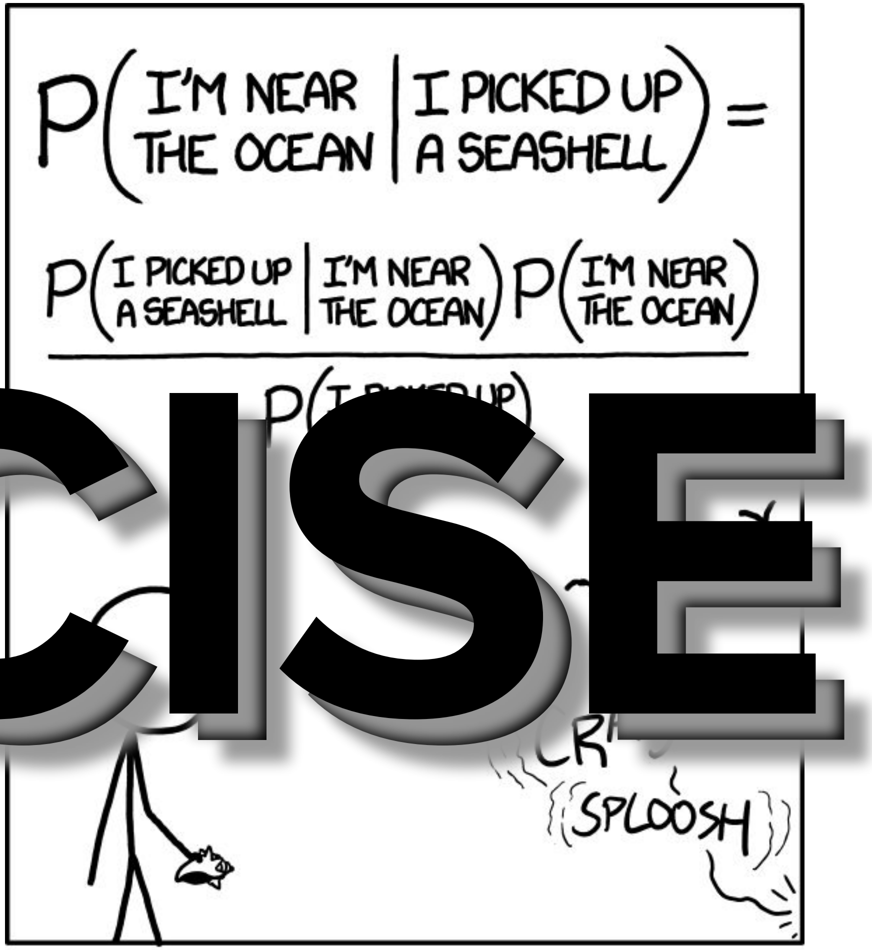
- Is it warm outside?
- Yes.
- Is it cold outside?
- *Are you an idiot!?*

Naïve Bayes Algorithm

“Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' th[illegible]ith s[illegible]ng “naï[illegible]” ind[illegible]e a[illegible]tions be[illegible]ea[illegible]s.

Na[illegible]s a p[illegible]metho[illegible]t th[illegible]blem of j[illegible]ume[illegible]s belong[illegible]to ca[illegible]a[illegible](su[illegible]s s[illegible]ti[illegible], et[illegible] 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.”

EXERCISE



P(I'M NEAR THE OCEAN | I PICKED UP A SEASHELL) =
P(I PICKED UP A SEASHELL | I'M NEAR THE OCEAN) P(I'M NEAR THE OCEAN)
P(I PICKED UP
CR
SPLOOSH

STATISTICALLY SPEAKING, IF YOU PICK UP A SEASHELL AND *DON'T* HOLD IT TO YOUR EAR, YOU CAN PROBABLY HEAR THE OCEAN.

DATASTAX®

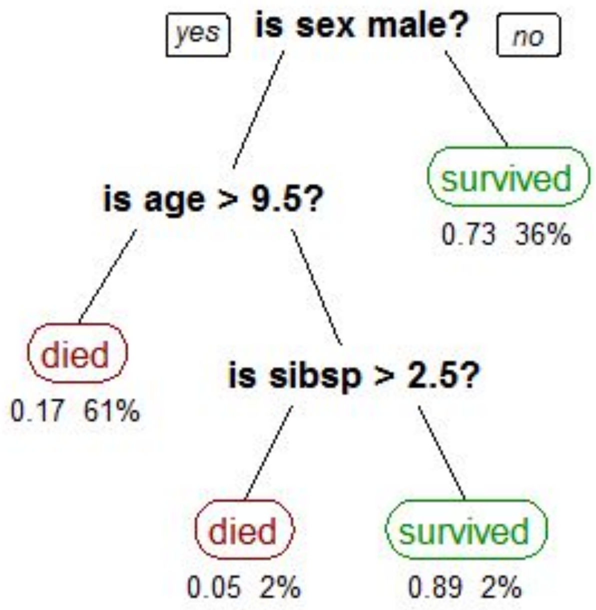
Random Forest

Supervised Classification Ensemble Method

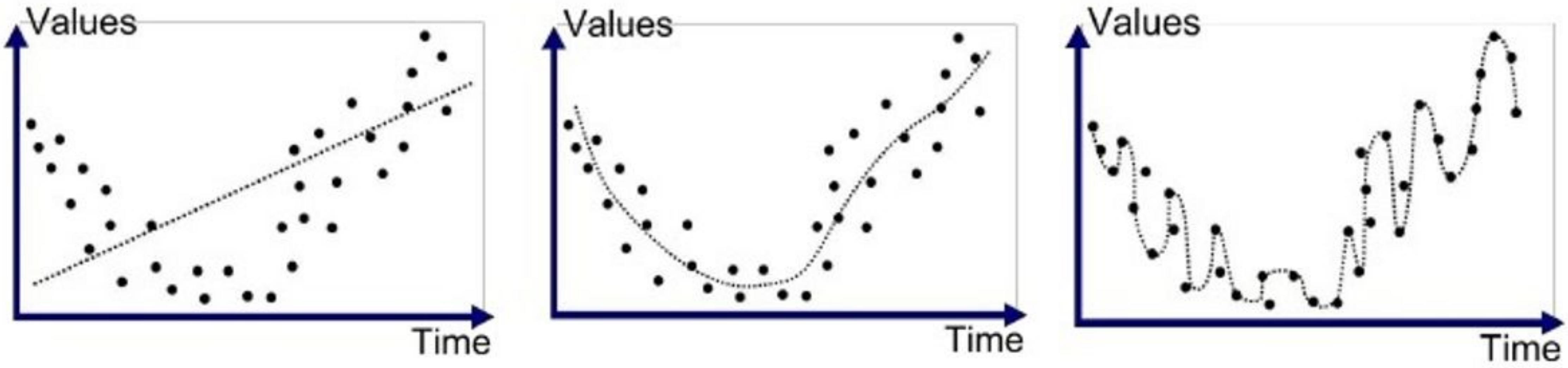
Decision Tree Algorithm

“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



Under-fitted vs Over-fitted



Underfitted

Good Fit/Robust

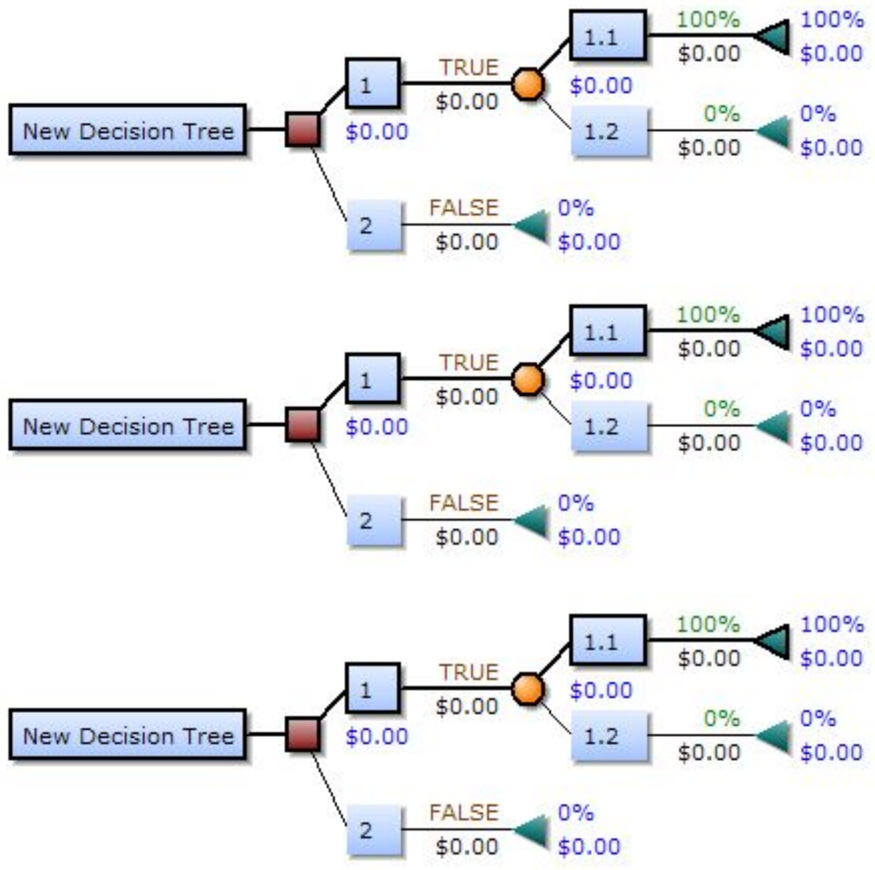
Overfitted

Anub Bhande @ Medium.com

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.”

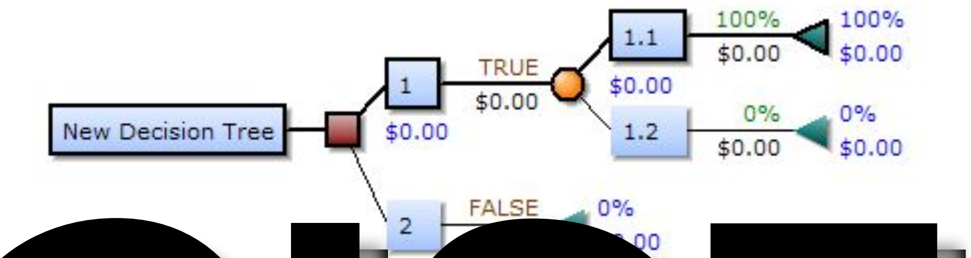


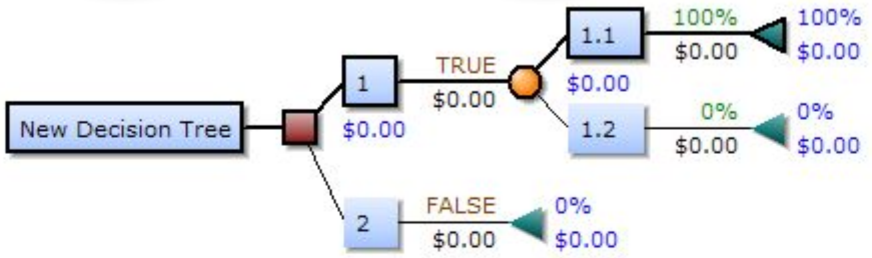
Random Forest Ensemble Method

“Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by co[illegible] m[illegible]itude [illegible] t[illegible] ti[illegible] outputti[illegible] th[illegible] the mod[illegible] the cl[illegible]ssificatio[illegible] mean [illegible] (re[illegible] of [illegible]al t[illegible]dom [illegible] co[illegible] for decis[illegible] e[illegible]bit[illegible]rfitting [illegible] ei[illegible] tra[illegible]

EXERCISE

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.”





DATASTAX®

Thank You!
Questions?