Decision Trees on Spark

Joseph K. Bradley

Decision Trees

Example: Spam detection

Get Viagra real cheap!
Send money now to get…

Goal:

Given an instance,
Examine its features, &
Predict a **label**

Not spam **Spam**

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False

Feature

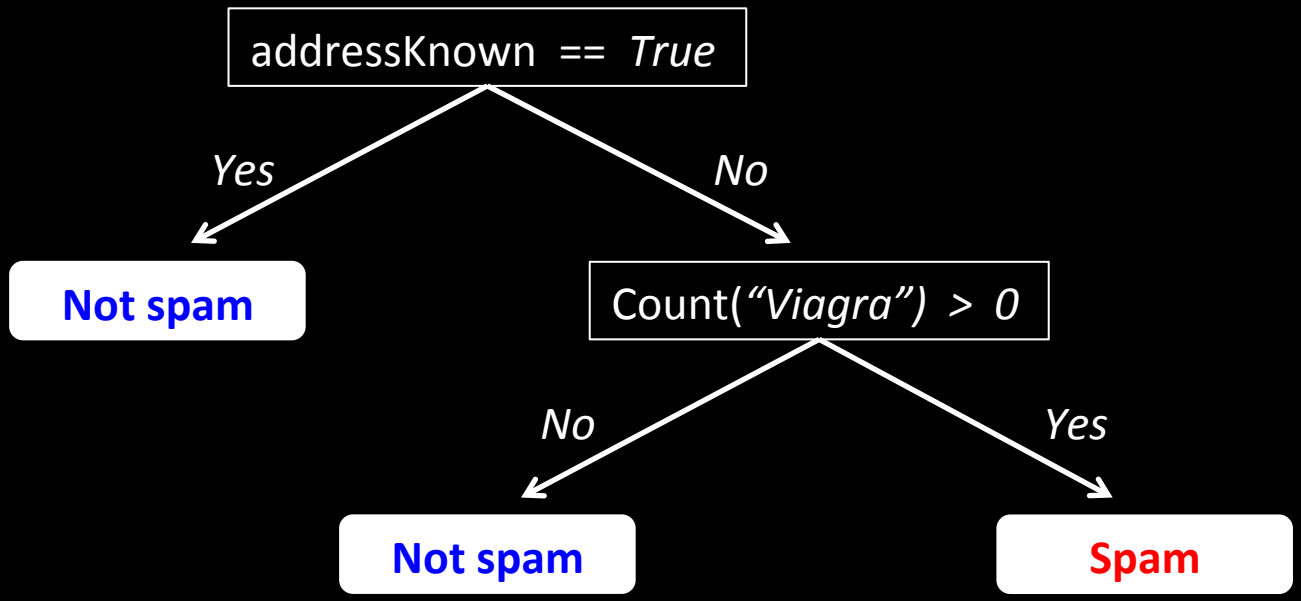
Instance/
example

databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False



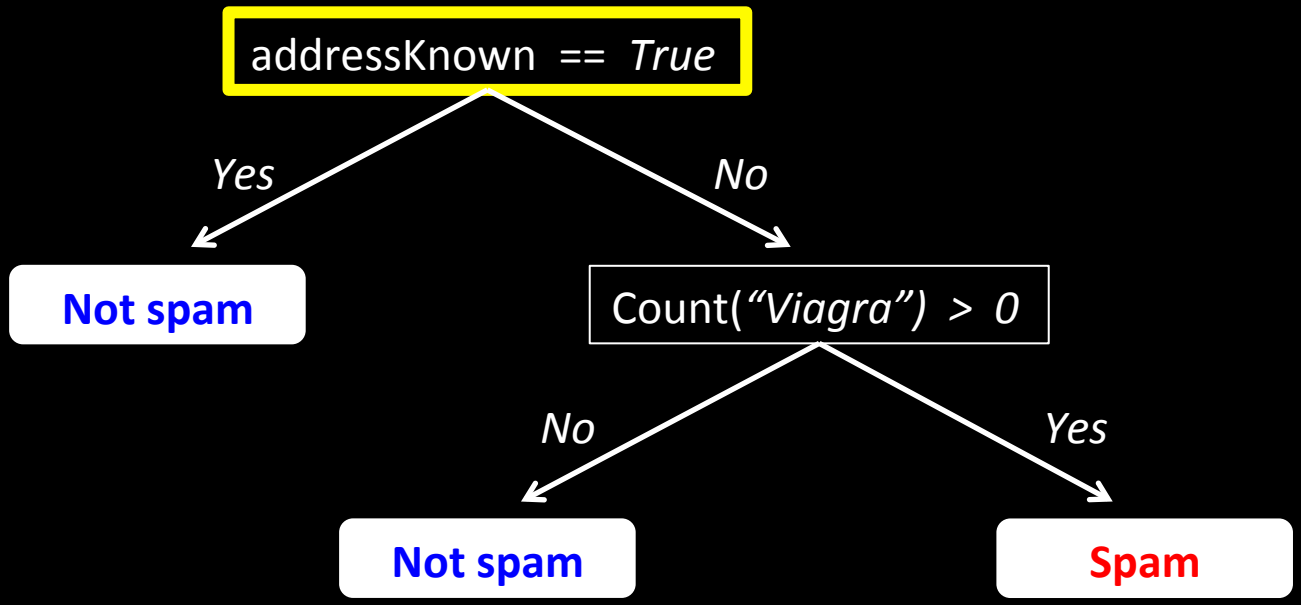
databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…
count	2	1	1	1	1	…

addressKnown
False

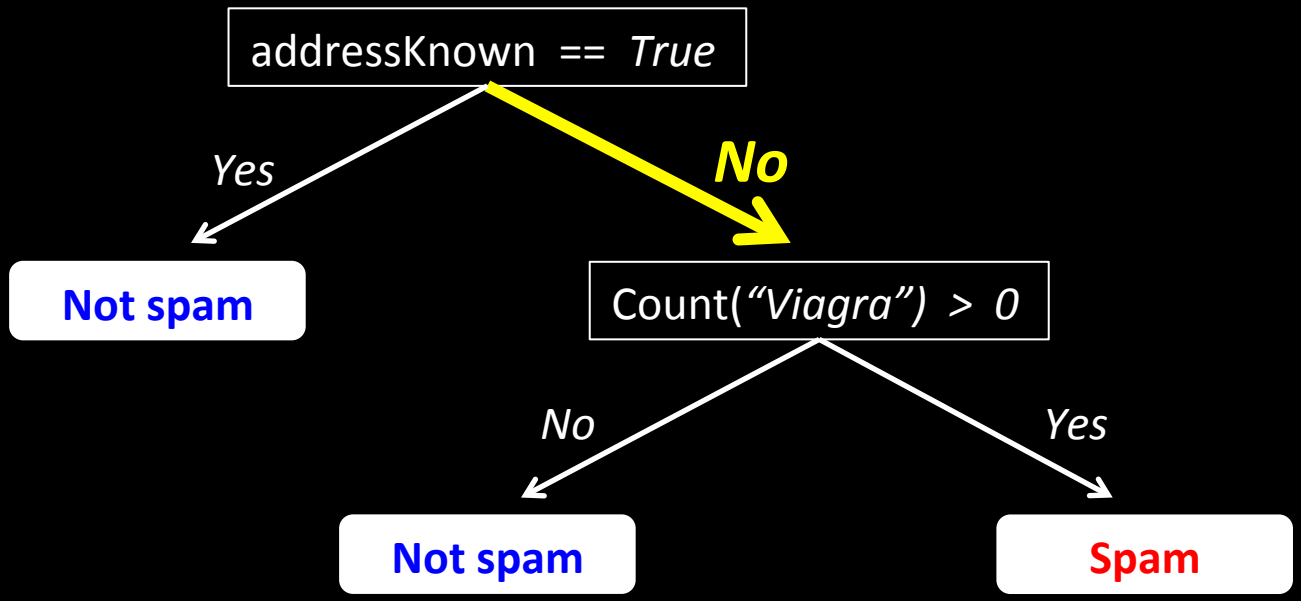


databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False

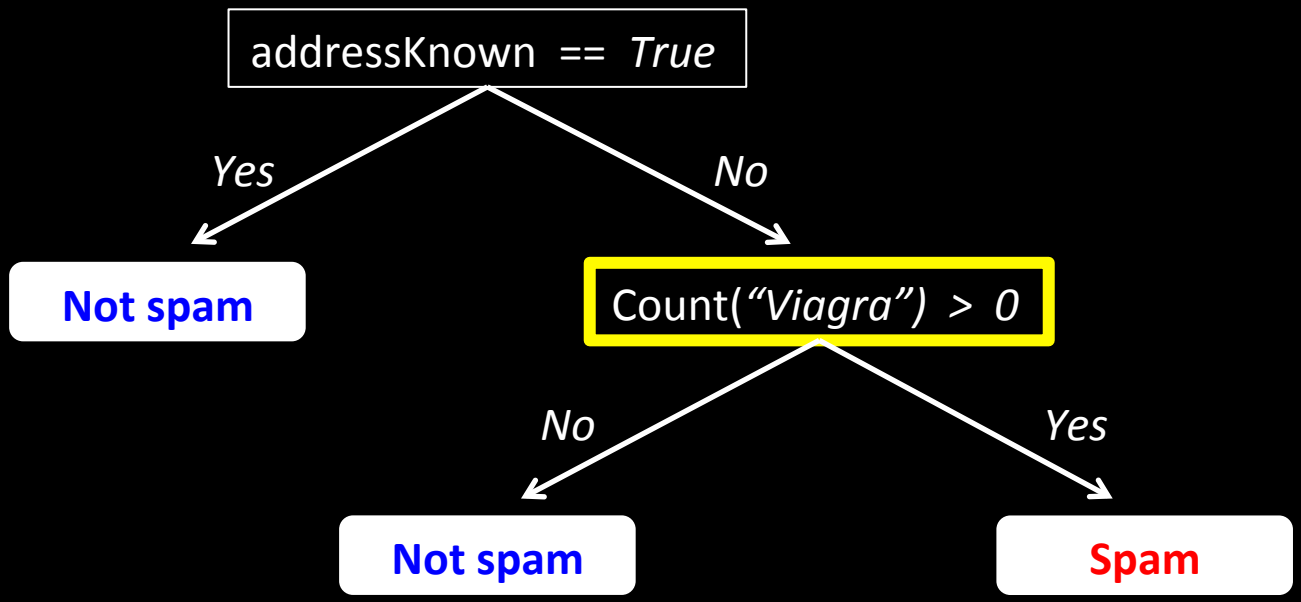


databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False

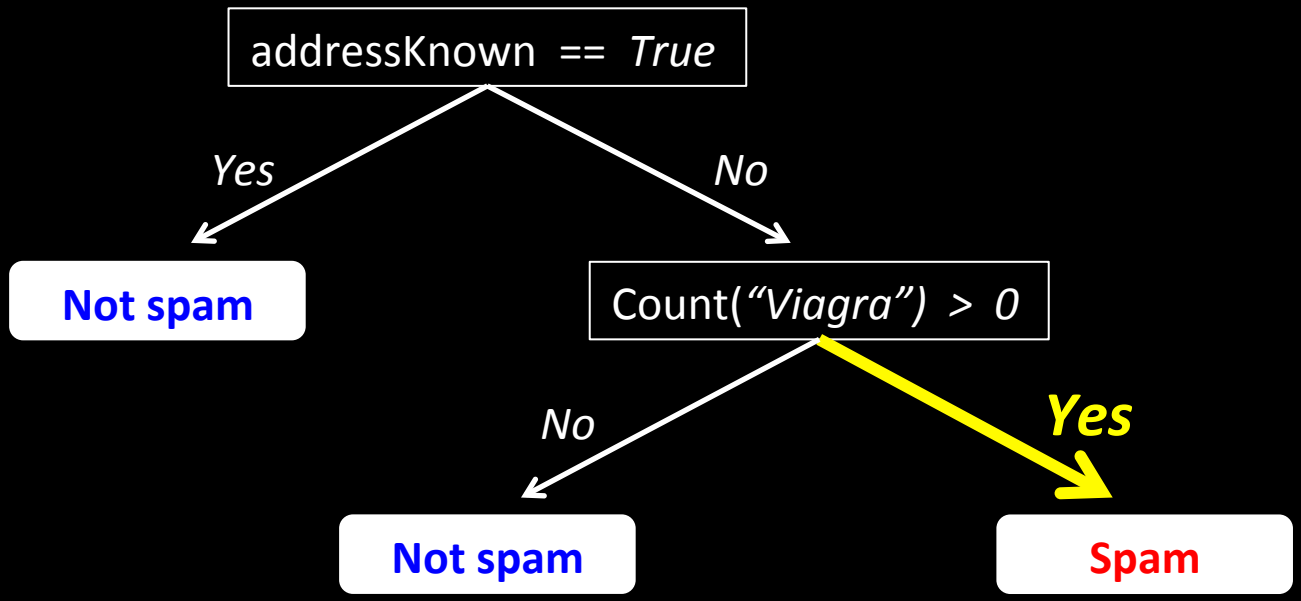


databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False

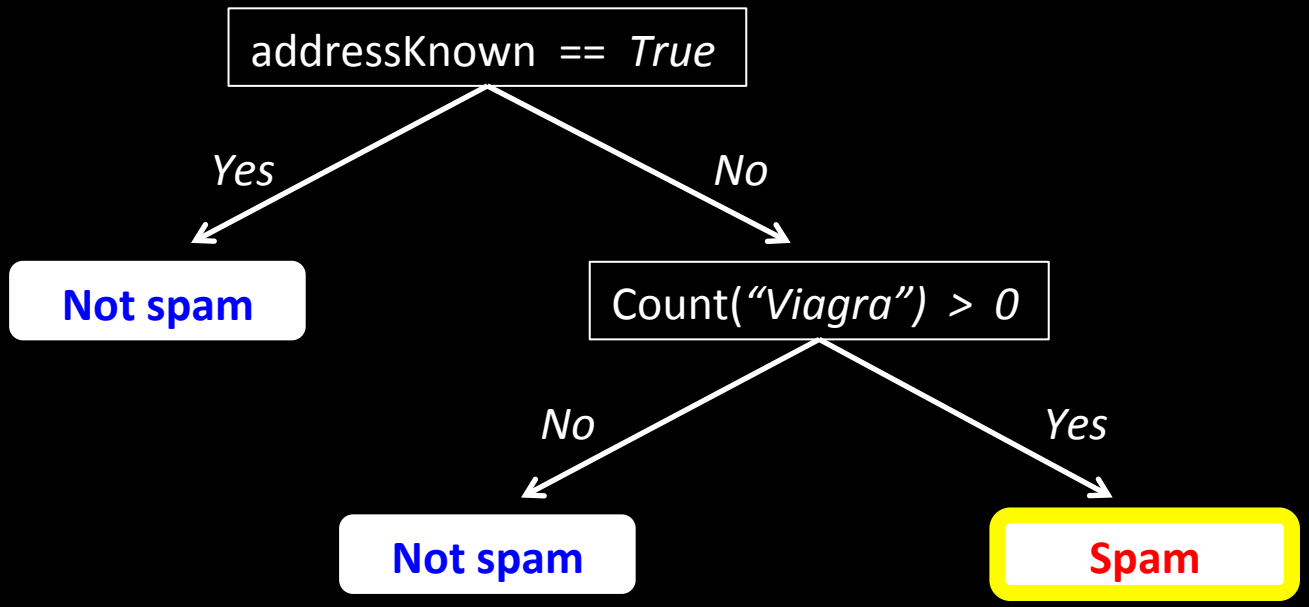


databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False

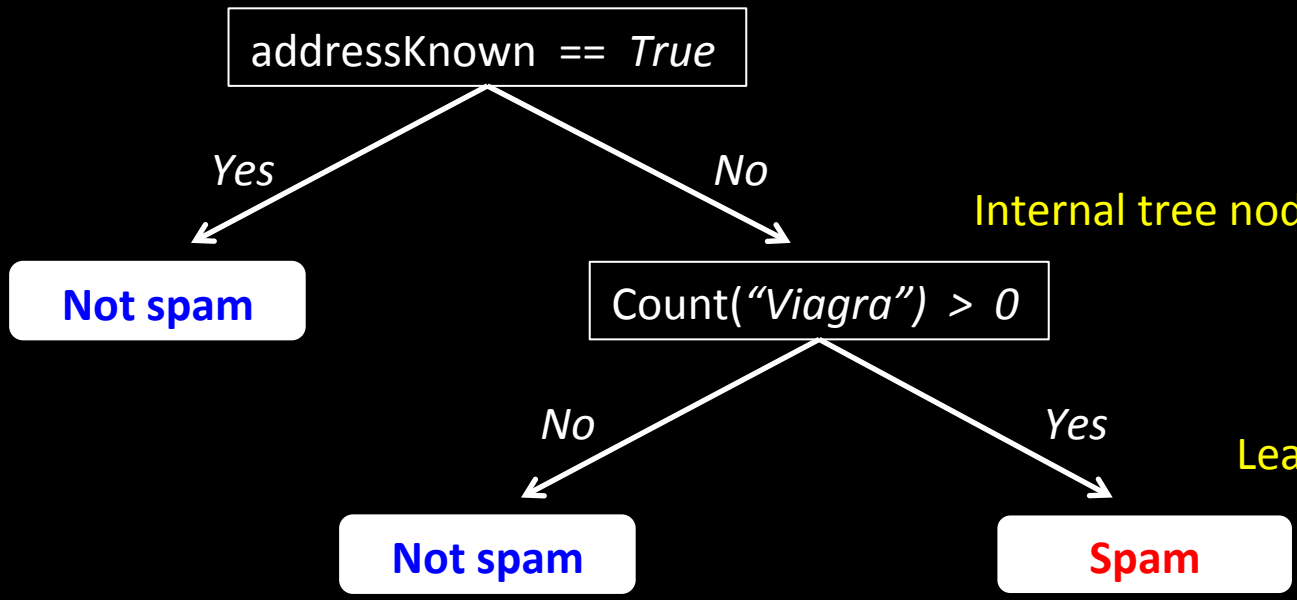


databricks

Decision Trees

Given: instance (feature vector)

word	get	Viagra	real	cheap	send	…	addressKnown
count	2	1	1	1	1	…	False



Internal tree node tests a feature

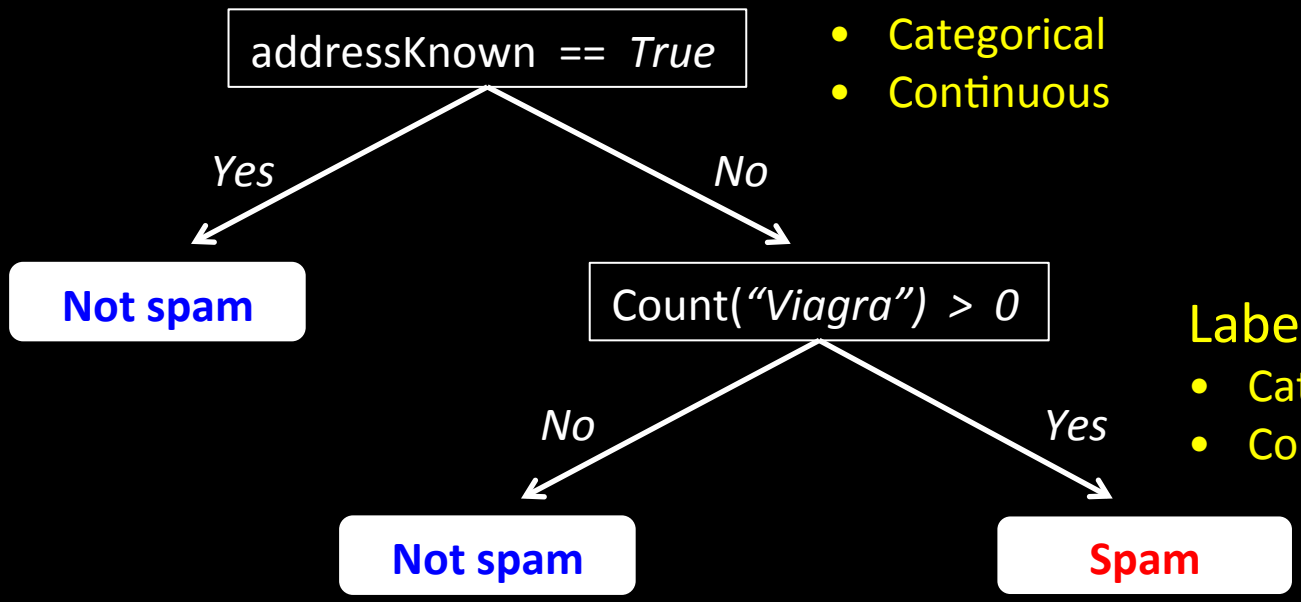
Leaf node predicts a label

databricks

Decision Trees

- Interpretable models
- Models can be simple (small) or expressive (big)

These are industry workhorses!



Features:

- Categorical
- Continuous

Labels:

- Categorical (classification)
- Continuous (regression)

databricks

Outline

- Decision Trees & Spark
- Learning Trees on Spark
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development

Outline

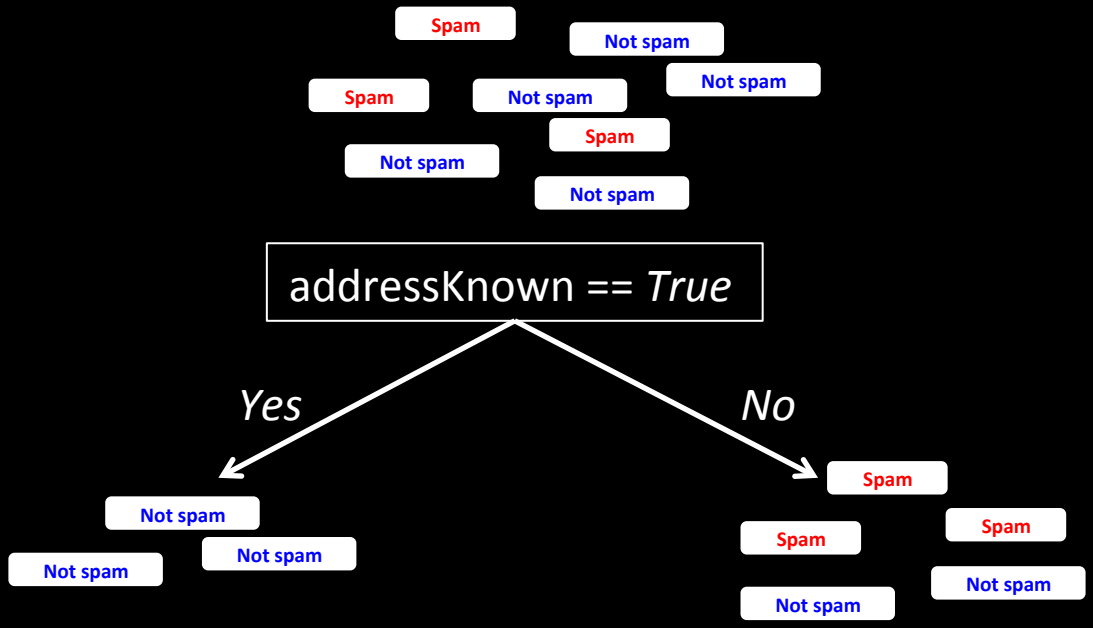
- **Decision Trees & Spark**
- Learning Trees on Spark
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development

Learning Decision Trees

Training data:

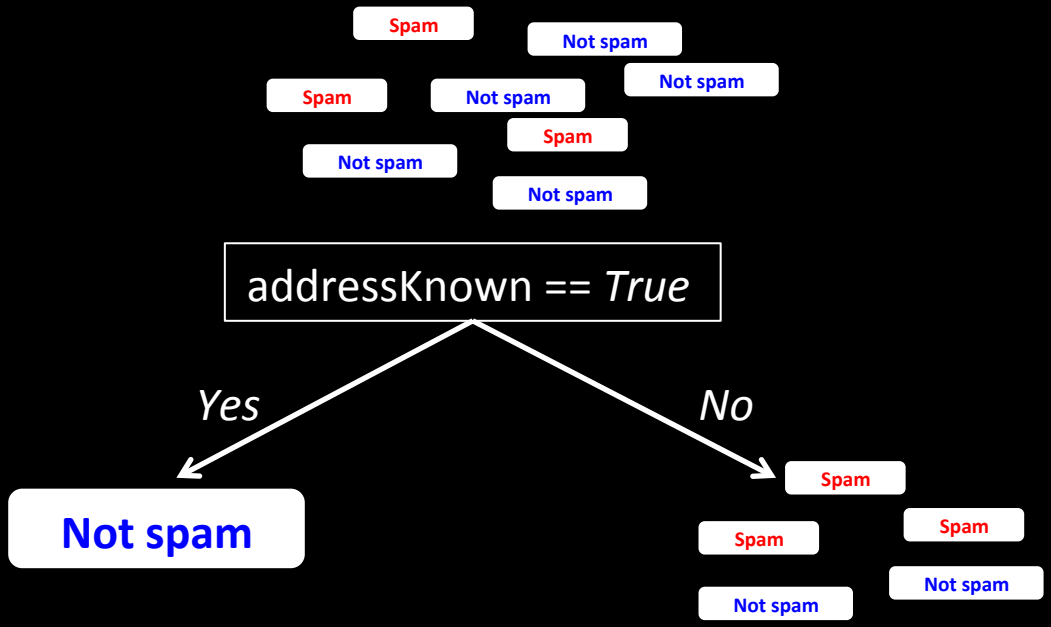
Spam	Get Viagra real cheap! Send money now to…
Not spam	Hi! I haven't seen you since the party last…
Not spam	Here's the info I promised to send you for…
Spam	Your password has been compromised! Go to…

Learning Decision Trees



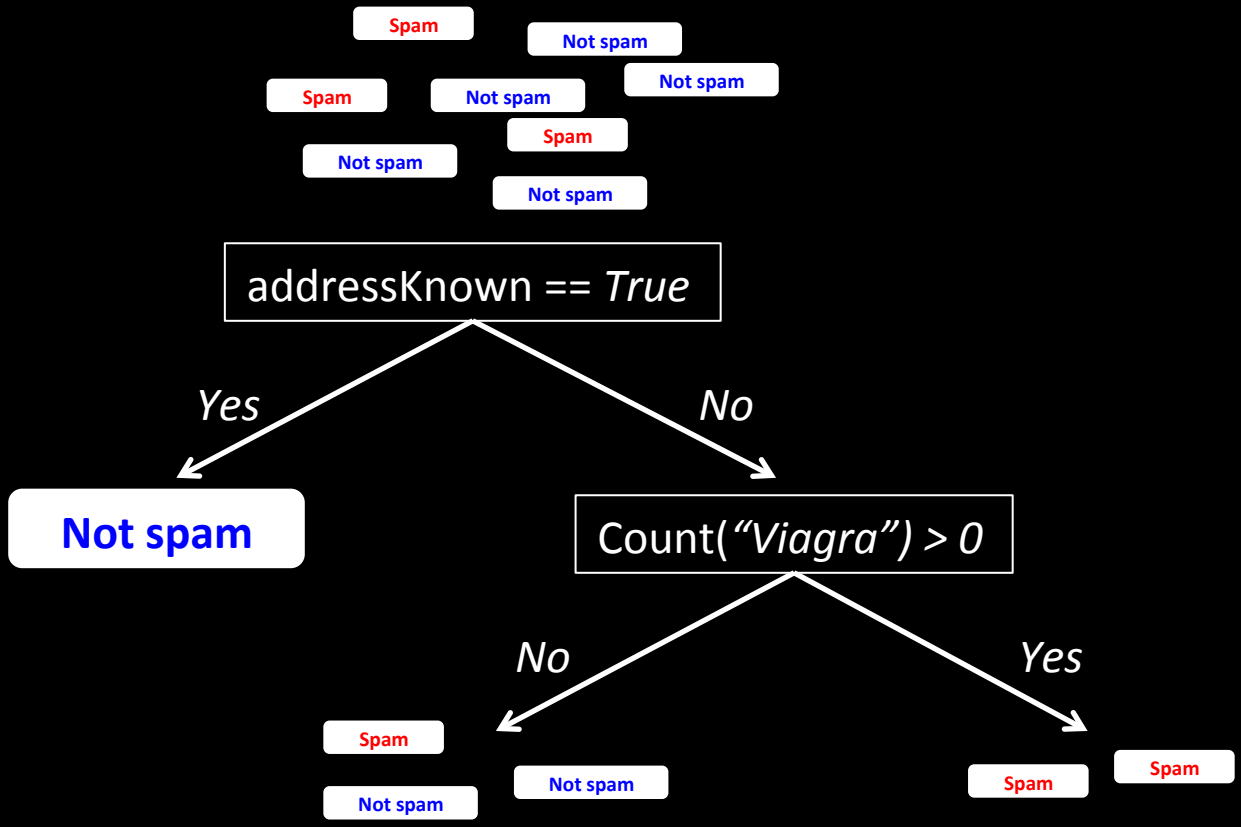
databricks

Learning Decision Trees



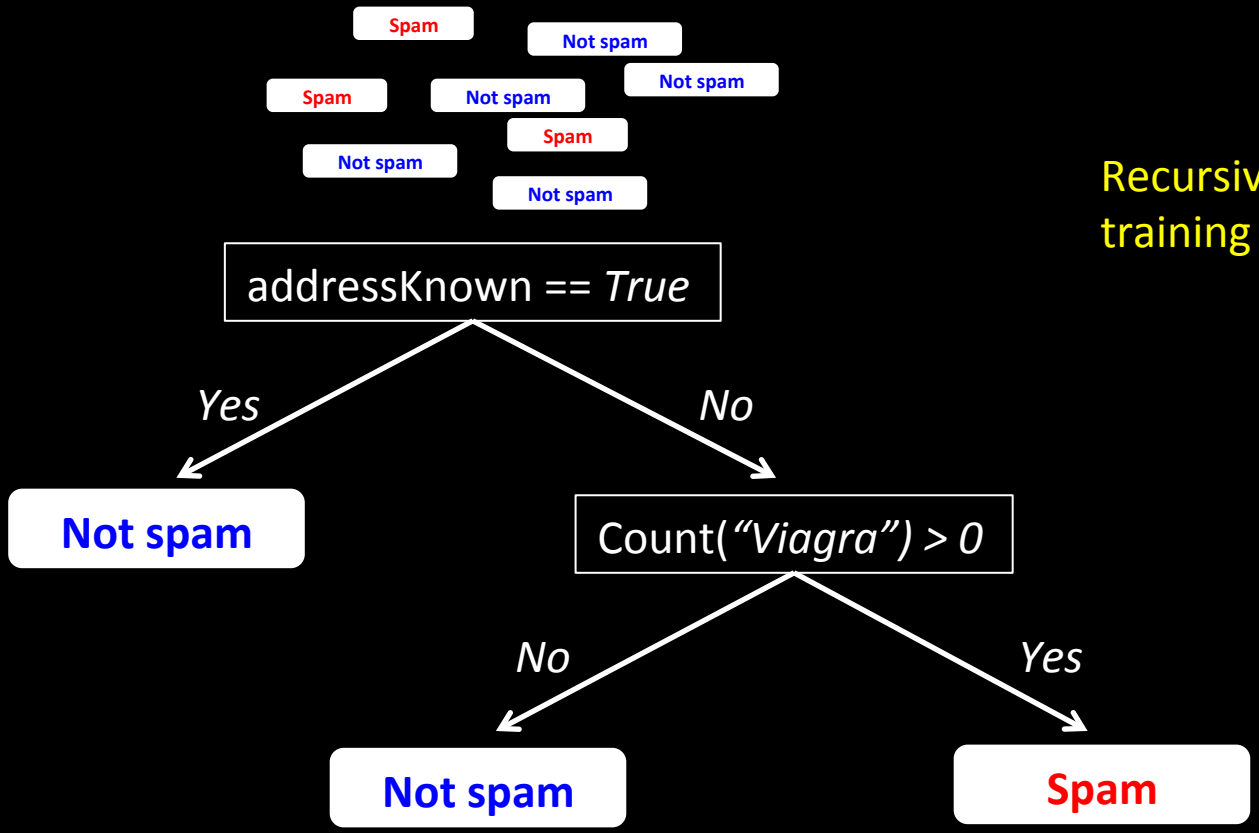
databricks

Learning Decision Trees



databricks

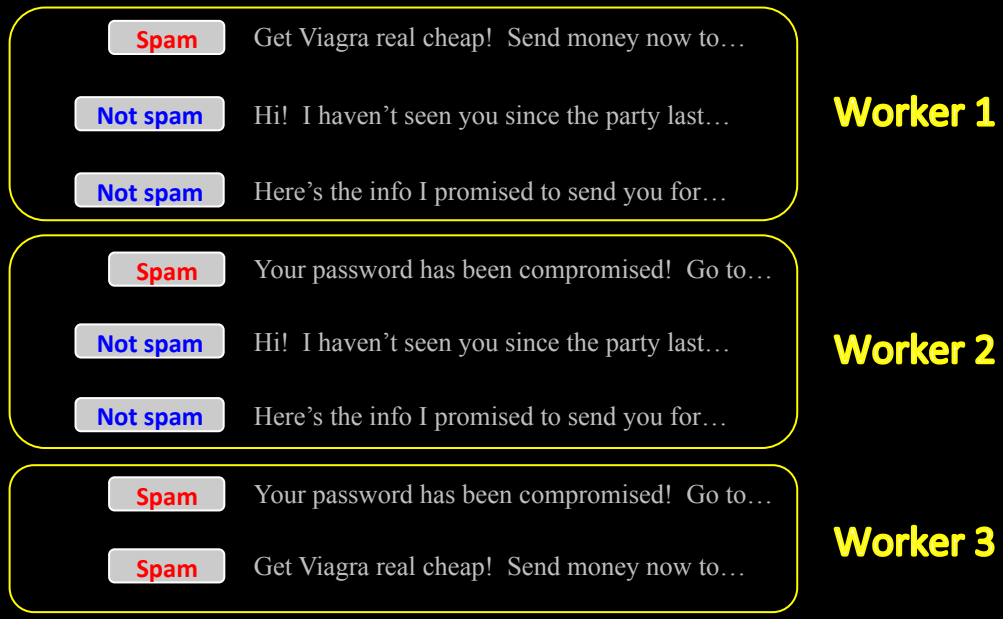
Learning Decision Trees



Recursively partition training instances

Distributed Learning of Trees

Distribute data: Partition by rows (instances)

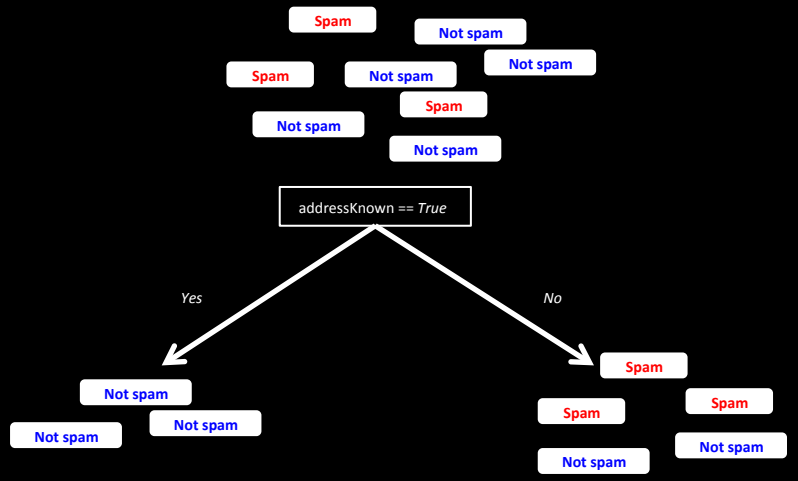


databricks

Distributed Learning of Trees

Distribute data: Partition by rows (instances)

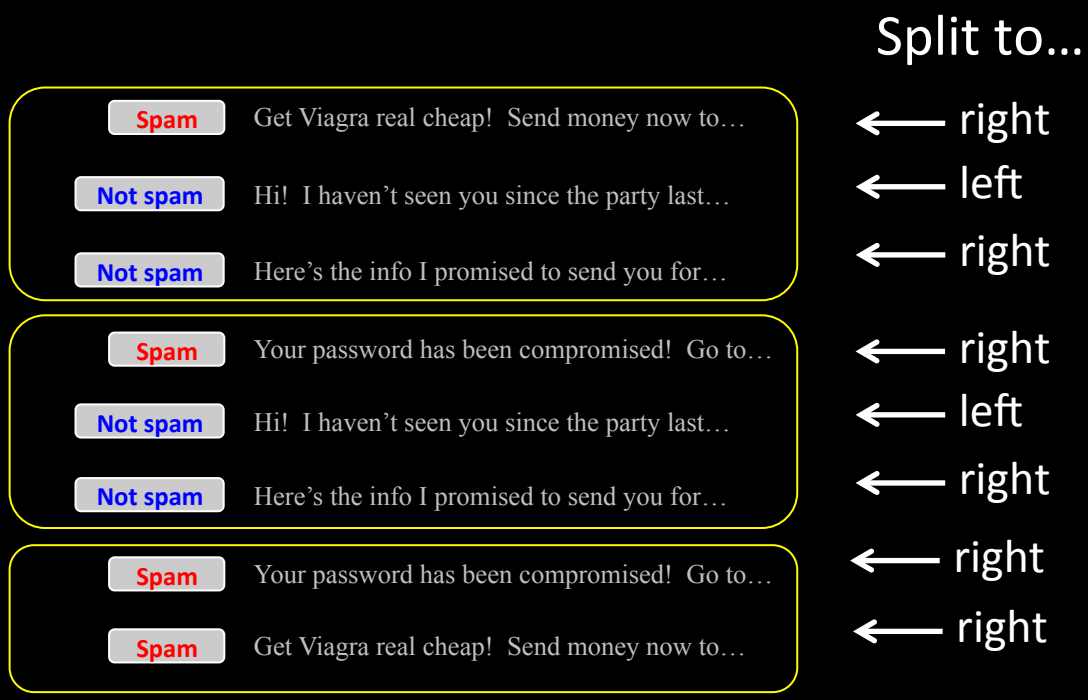
Spam	Get Viagra real cheap! Send money now to…
Not spam	Hi! I haven't seen you since the party last…
Not spam	Here's the info I promised to send you for…
Spam	Your password has been compromised! Go to…
Not spam	Hi! I haven't seen you since the party last…
Not spam	Here's the info I promised to send you for…
Spam	Your password has been compromised! Go to…
Spam	Get Viagra real cheap! Send money now to…



databricks

Distributed Learning of Trees

Distribute data: Partition by rows (instances)



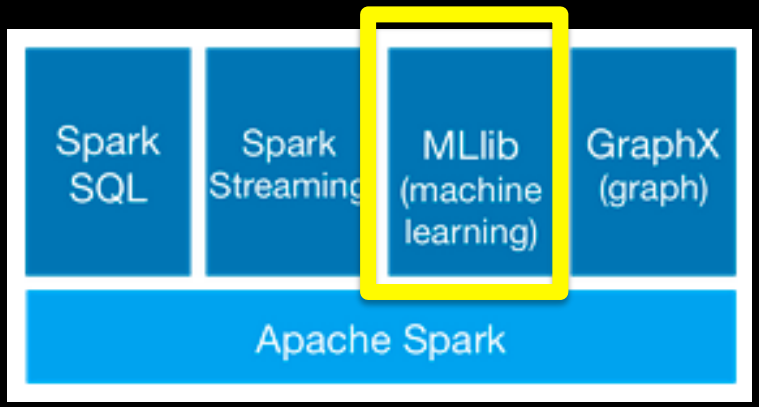
Traditional algorithm:
Shuffle entire dataset across network many times ☹

databricks

Spark

Fast & general engine for large-scale data processing

- Started by UC Berkeley AMPLab in 2009
- Big open source community
  - One of fastest growing Apache projects
  - 250+ developers from 50 companies
- Included in major Hadoop distributions



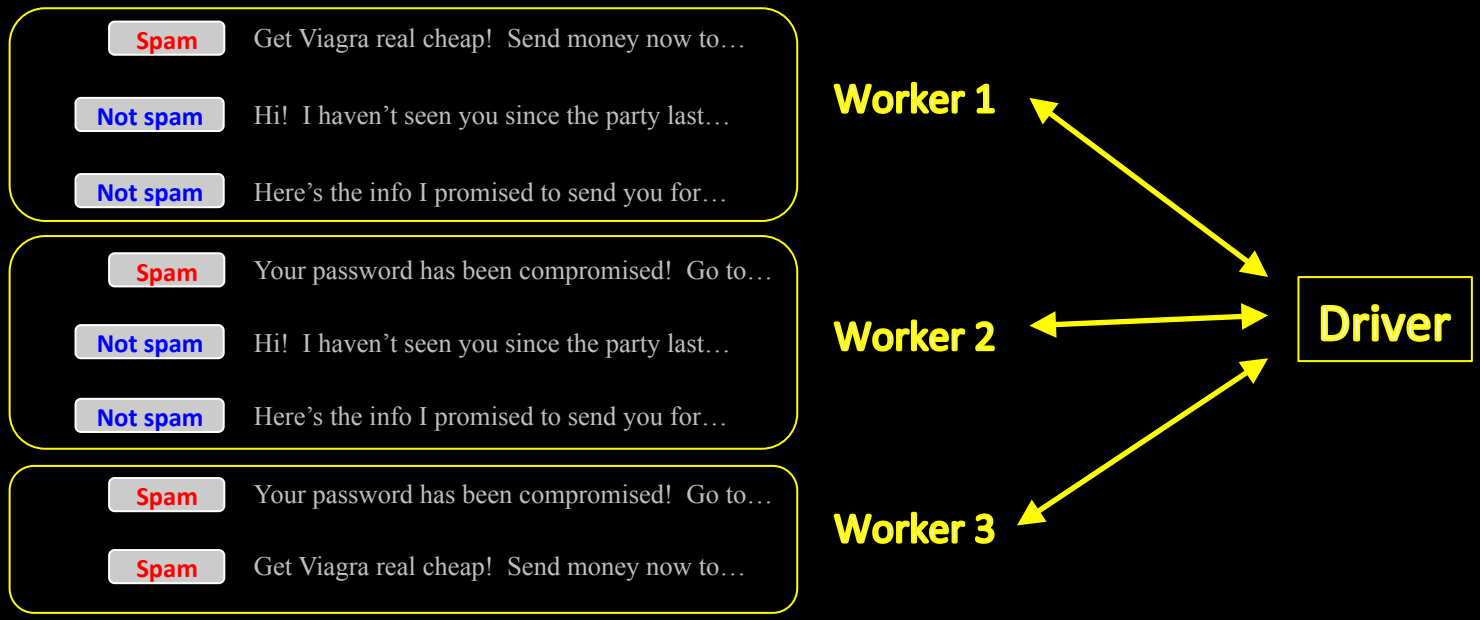
Mllib

- Classification
- Regression
- Recommendation
- Clustering
- Statistics
- Linear algebra

databricks

Spark RDDs

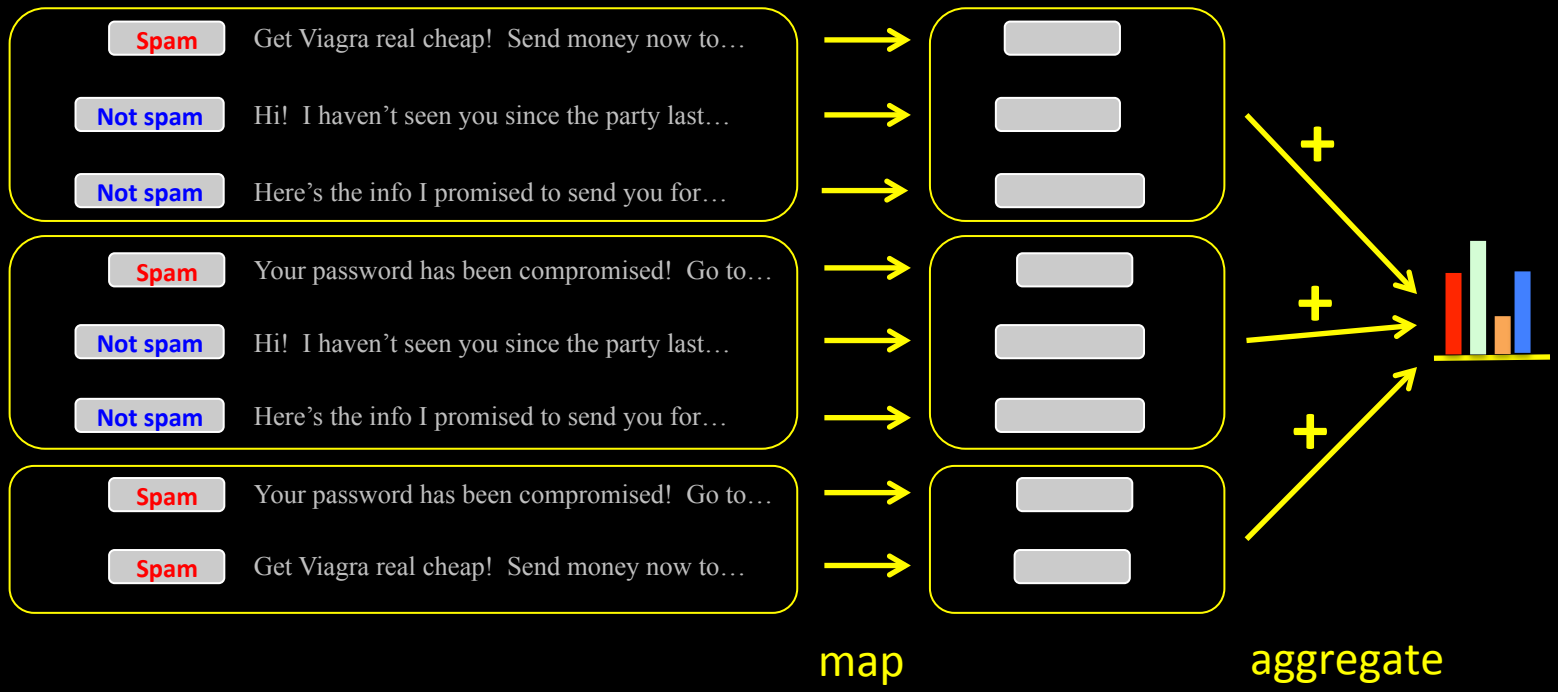
Resilient Distributed Datasets (RDDs)



databricks

Spark RDDs

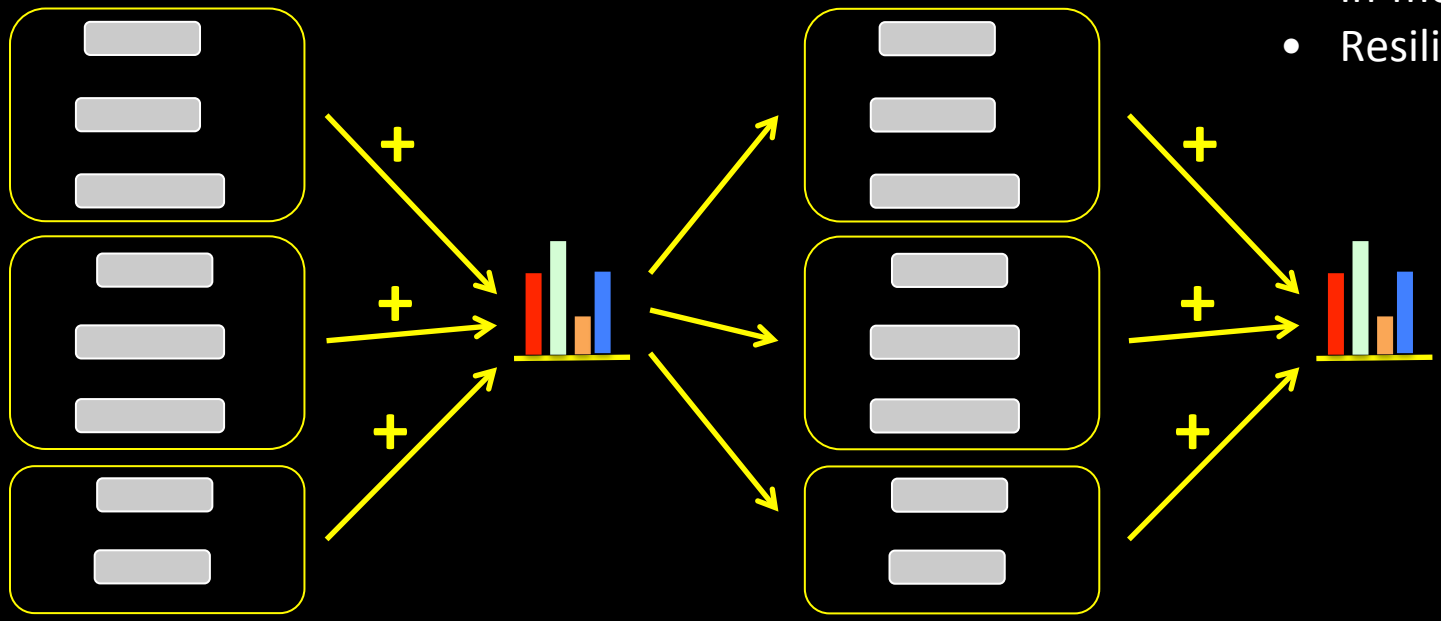
Resilient Distributed Datasets (RDDs)



databricks

Iterative Computation on Spark

RDDs
- In-memory
- Resilient to failures



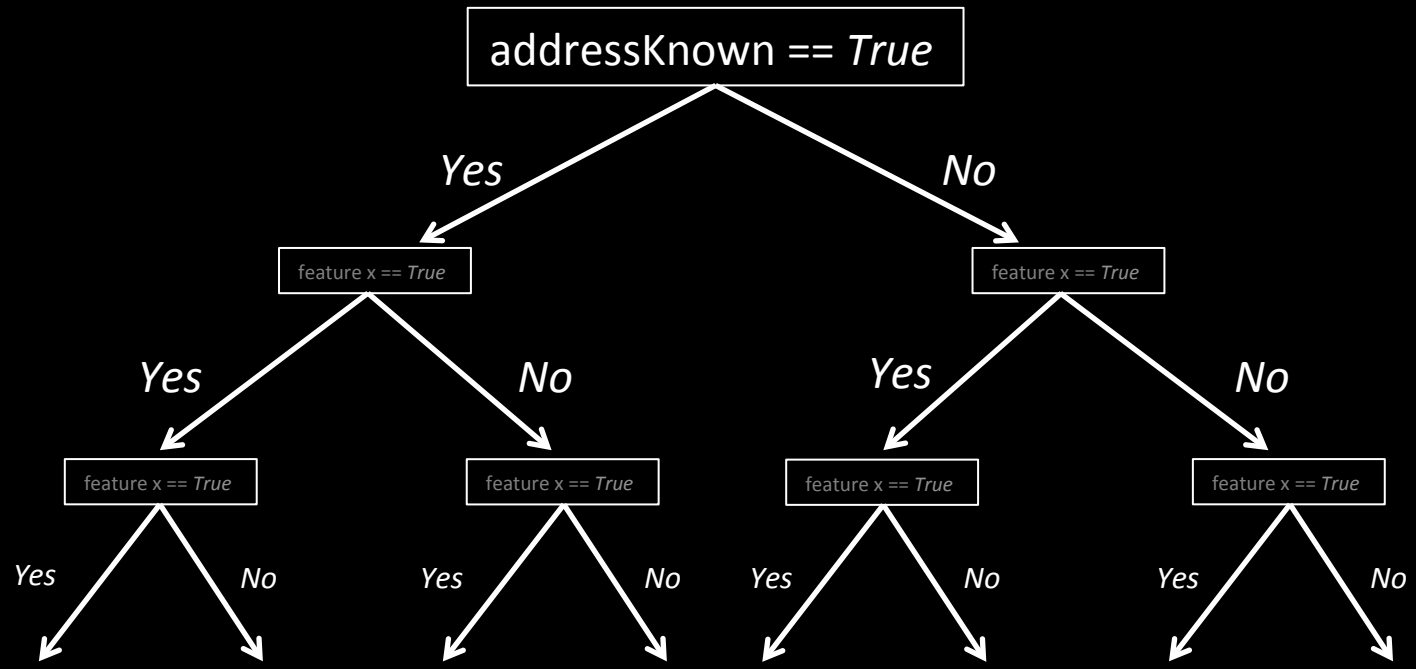
Outline

- **Decision Trees & Spark**
- Learning Trees on Spark
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development

Outline

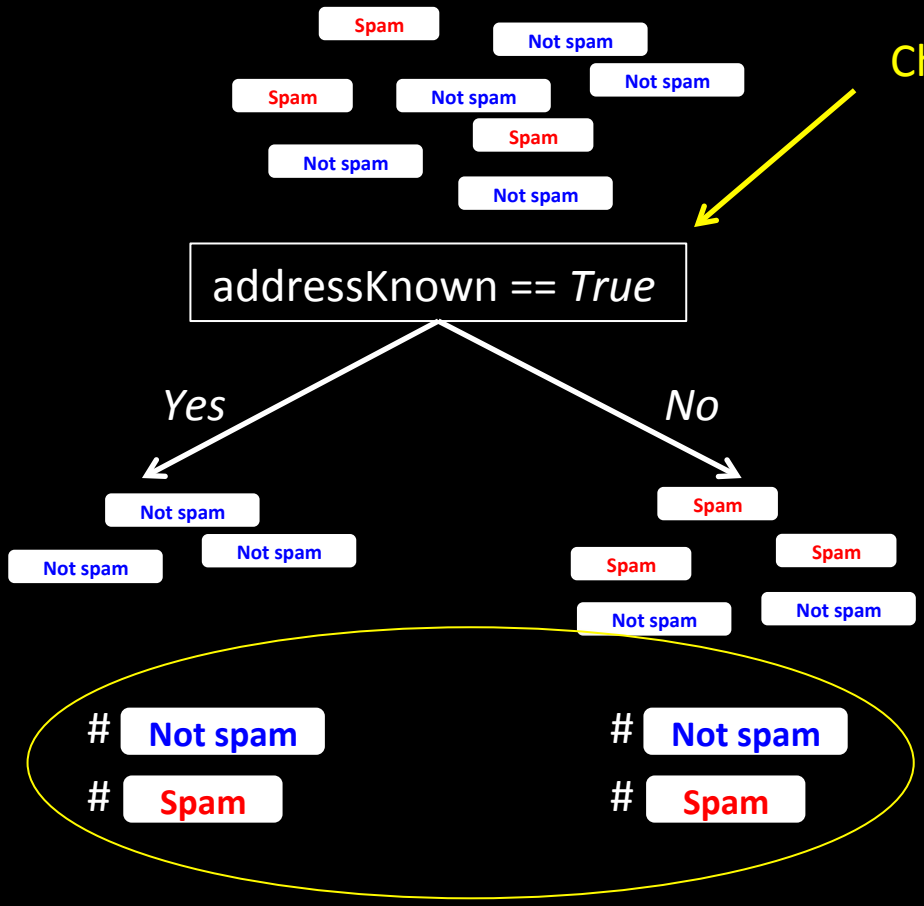
- Decision Trees & Spark
- **Learning Trees on Spark**
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development

Train Level-by-Level



databricks

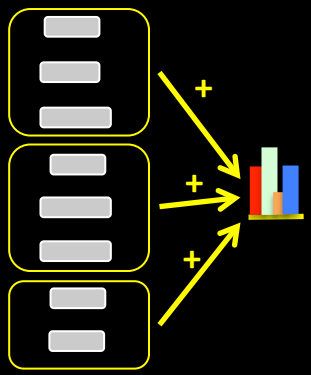
Choosing How to Split



Choose 1 feature x_j to test:

Binary feature: x_j == True/False

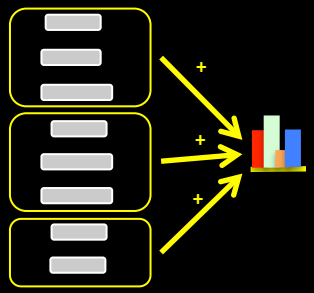
How good is this split?



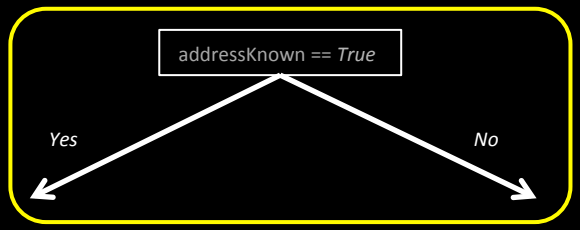
Sufficient statistics
for this split
→ compute **impurity** (information gain)

databricks

High-Level View

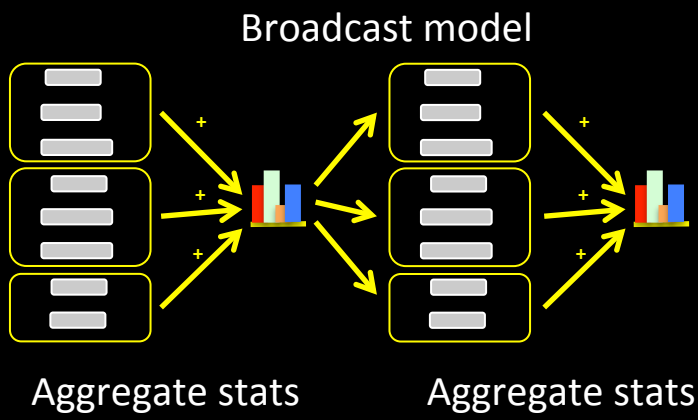


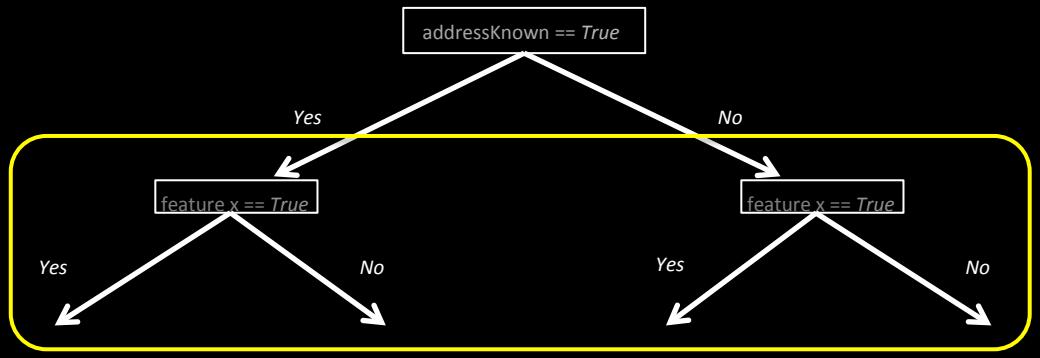
Aggregate stats



databricks

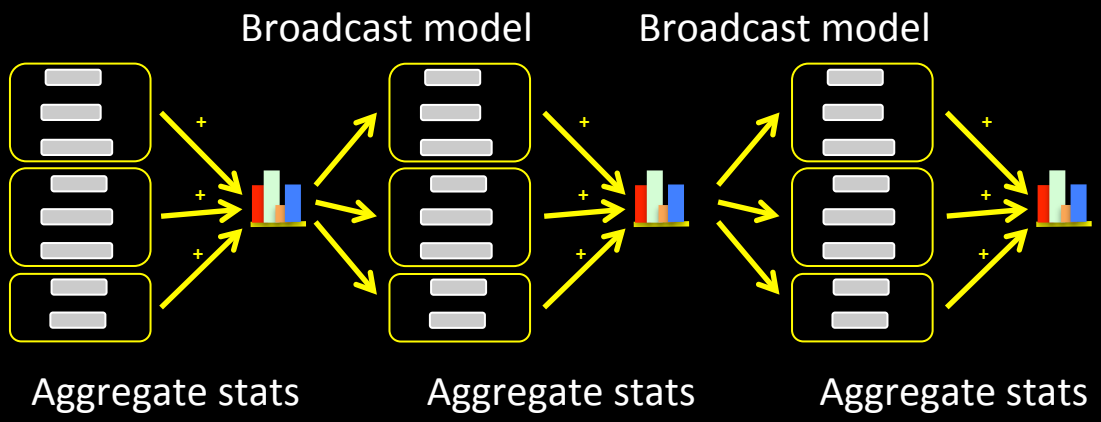
High-Level View





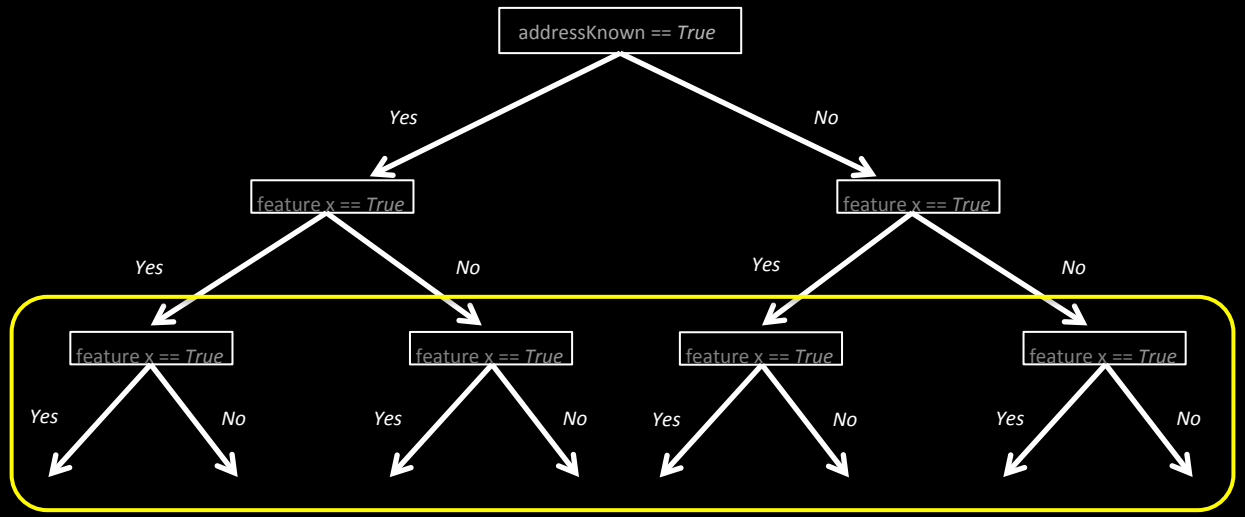
databricks

High-Level View



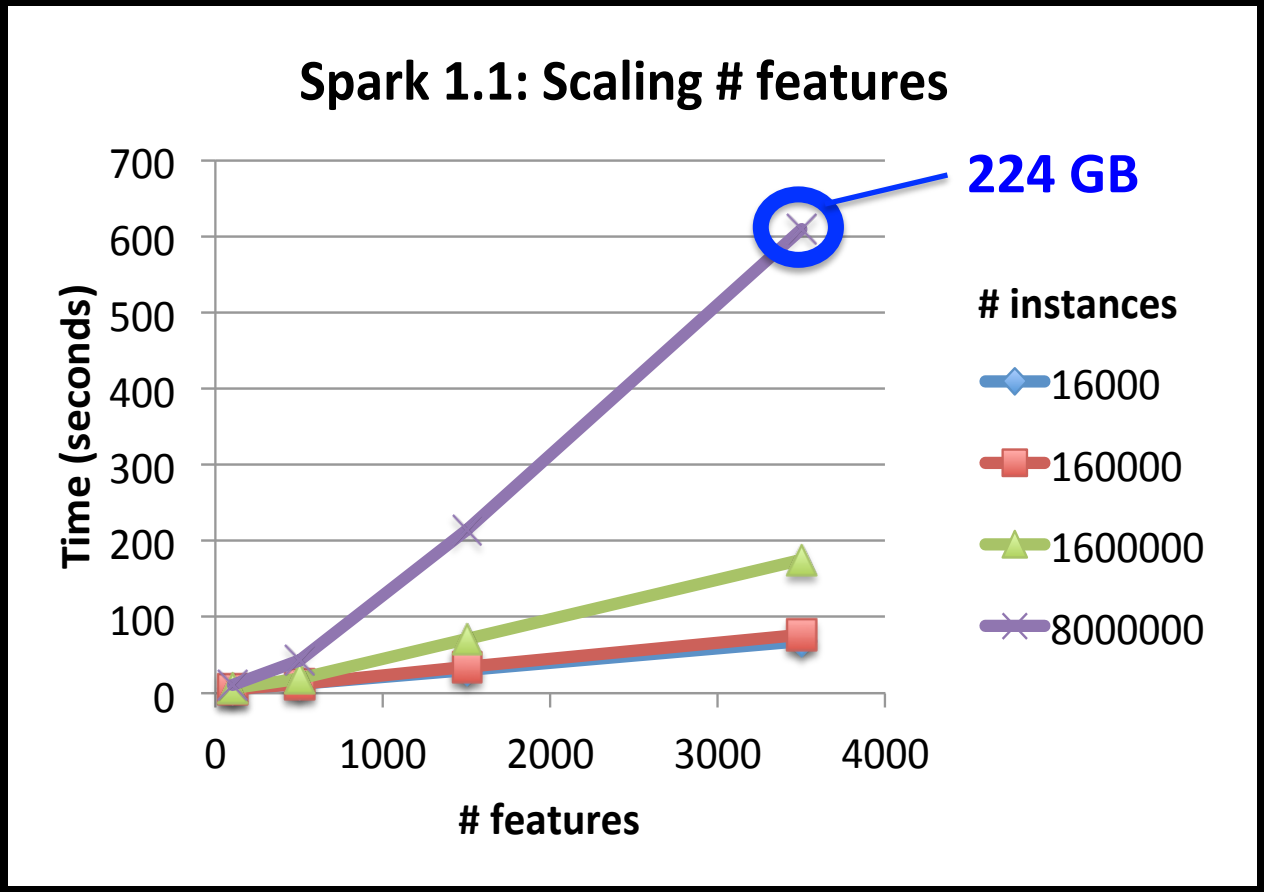
Only 1 pass over data per level

➔ Running time scales linearly with dataset size.



databricks

Scaling with Dataset Size

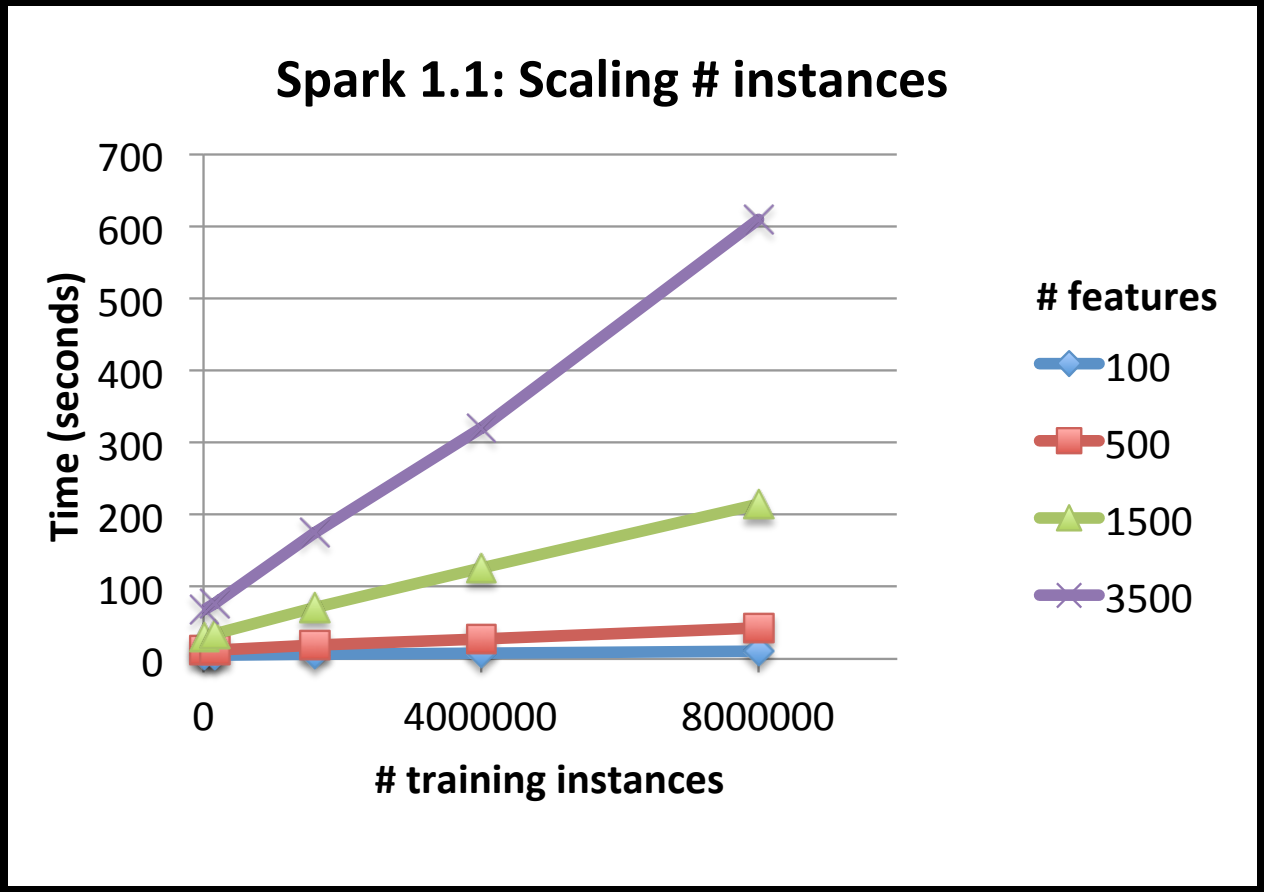


Binary classification
16-node EC2
6-level trees

Runtime scales linearly with # features

databricks

Scaling with Dataset Size



Binary classification
16-node EC2
6-level trees

Runtime scales linearly with # instances

databricks

Outline

- Decision Trees & Spark
- **Learning Trees on Spark**
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development

Outline

- Decision Trees & Spark
- Learning Trees on Spark
- **Using MLlib Trees in Practice**
  - **Model selection**
  - **Accuracy—communication trade-offs**
- Active Development

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

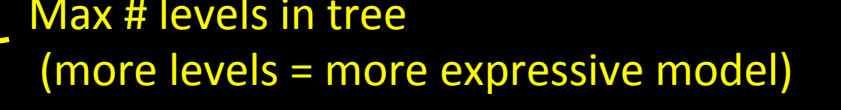
MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

Measures how good a split is.
(information gain)

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```



databricks

(demo: maxDepth & impurity)

DecisionTree

```
> import org.apache.spark.mllib.util.MLUtils
  import org.apache.spark.mllib.tree.DecisionTree
  import org.apache.spark.mllib.tree.model.DecisionTreeModel
```

```
import org.apache.spark.mllib.util.MLUtils
import org.apache.spark.mllib.tree.DecisionTree
import org.apache.spark.mllib.tree.model.DecisionTreeModel
```

Command took 0.14s

Choosing number of bins

Load training, test datasets

Command took 0.02s

```
> val mnistTrain = MLUtils.loadLibSVMFile(sc, "/mnt/meng-s3/mllib-
  data/mnist-digits/mnist-digits-train.txt")
  val mnistTest = MLUtils.loadLibSVMFile(sc, "/mnt/meng-s3/mllib-data/mnist-
  digits/mnist-digits-test.txt")
```

```
mnistTrain: org.apache.spark.rdd.RDD[org.apache.spark.mllib.regression.LabeledPoint] =
MappedRDD[4363] at map at MLUtils.scala:98
```

```
> (mnistTrain.count, mnistTest.count)
res4: (Long, Long) = (60000,10000)
Command took 0.20s
```

Train Decision Trees using maxDepth = 0,1,...,7, and impurity = Gini, Entropy

```
> val treesGini = (0 until 8).map { maxDepth =>
    DecisionTree.trainClassifier(
      mnistTrain,
      numClassesForClassification = 10,
      categoricalFeaturesInfo = Map.empty[Int, Int],
      impurity = "gini",
      maxDepth = maxDepth,
      maxBins = 4)
  }
  val treesEntropy = (0 until 8).map { maxDepth =>
    DecisionTree.trainClassifier(
      mnistTrain,
      numClassesForClassification = 10,
      categoricalFeaturesInfo = Map.empty[Int, Int],
      impurity = "entropy",
      maxDepth = maxDepth,
      maxBins = 4)
  }
```

```
treesGini: scala.collection.immutable.IndexedSeq[org.apache.spark.mllib.tree.model.Deci
sionTreeModel] =
Vector(DecisionTreeModel classifier
  Predict: 1.0
, DecisionTreeModel classifier
  If (feature 409 <= 0.0)
   Predict: 1.0
  Else (feature 409 > 0.0)
   Predict: 9.0
, DecisionTreeModel classifier
  If (feature 409 <= 0.0)
   If (feature 434 <= 0.0)
    Predict: 0.0
   Else (feature 434 > 0.0)
    Predict: 1.0
  Else (feature 409 > 0.0)
   If (feature 155 <= 0.0)
    Predict: 7.0
   Else (feature 155 > 0.0)
    Predict: 3.0
, DecisionTreeModel classifier
  If (feature 409 <= 0.0)
   If (feature 434 <= 26.0)
    If (feature 455 <= 0.0)
     Predict: 5.0
    Else (feature 455 > 0.0)
     Predict: 0.0
   Else (feature 434 > 26.0)
    If (feature 375 <= 0.0)
```

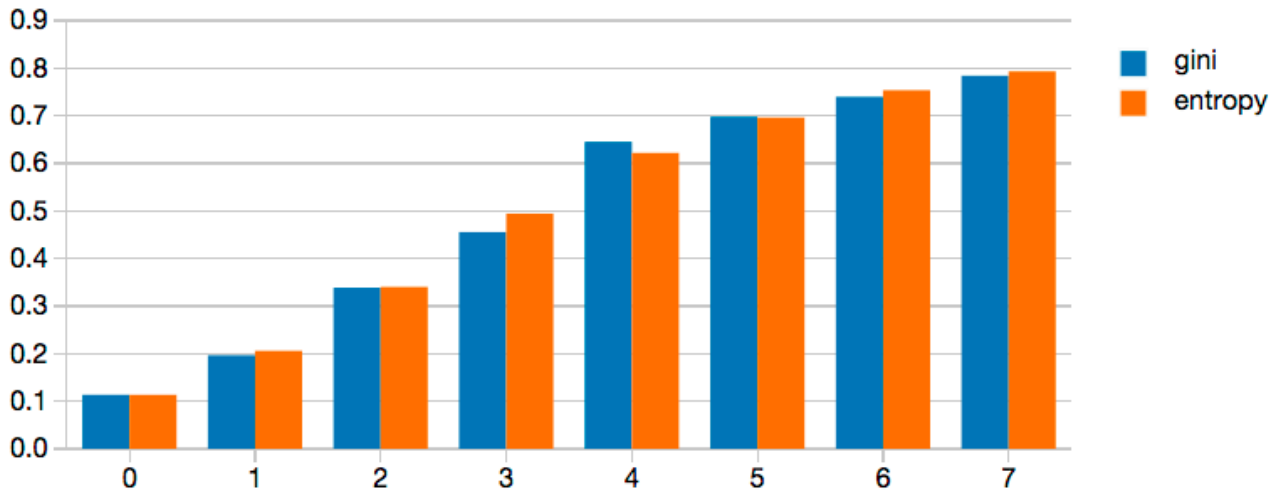
trees - Databricks

https://alpha.databricks.com/#shell/954043

trees (Scala)

Attached: Default Cluster Run All Arguments

```
> case class Accuracy(maxDepth: Int, impurity: String, accuracy: Double)
  val giniEntropyAccuracies = (0 until 8).flatMap { maxDepth =>
    Array(
      new Accuracy(maxDepth, "gini", accuracy(treesGini(maxDepth))),
      new Accuracy(maxDepth, "entropy", accuracy(treesEntropy(maxDepth)))
    )
  }
  display(sc.makeRDD(giniEntropyAccuracies))
```



Plot Options...

Command took 4.90s

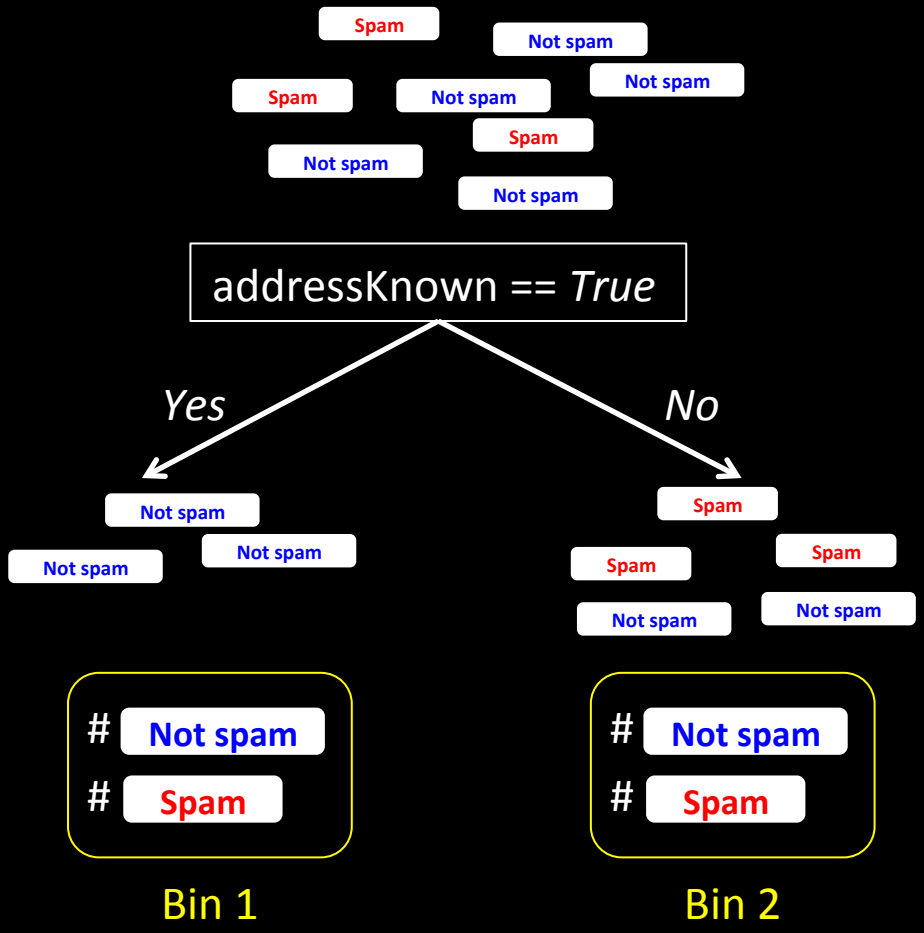
MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

Choosing How to Split



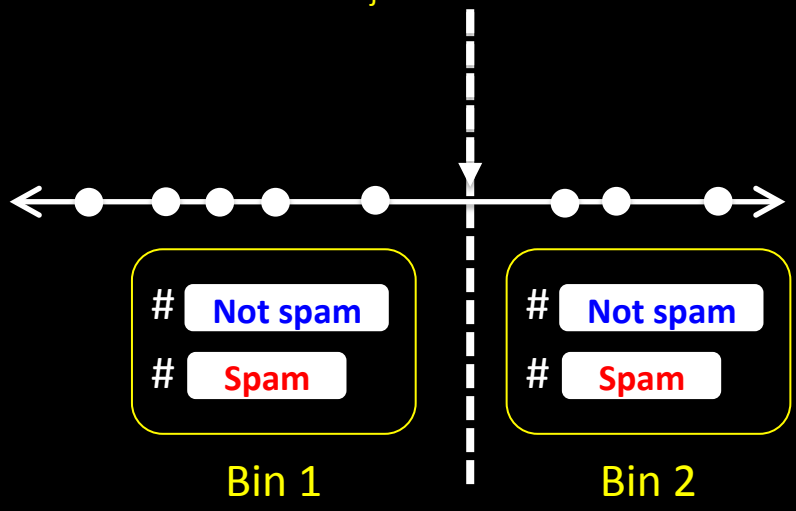
Binary feature: x_j == True/False

→ 2 bins / feature

databricks

Binning Features

Continuous feature: x_j < (value)



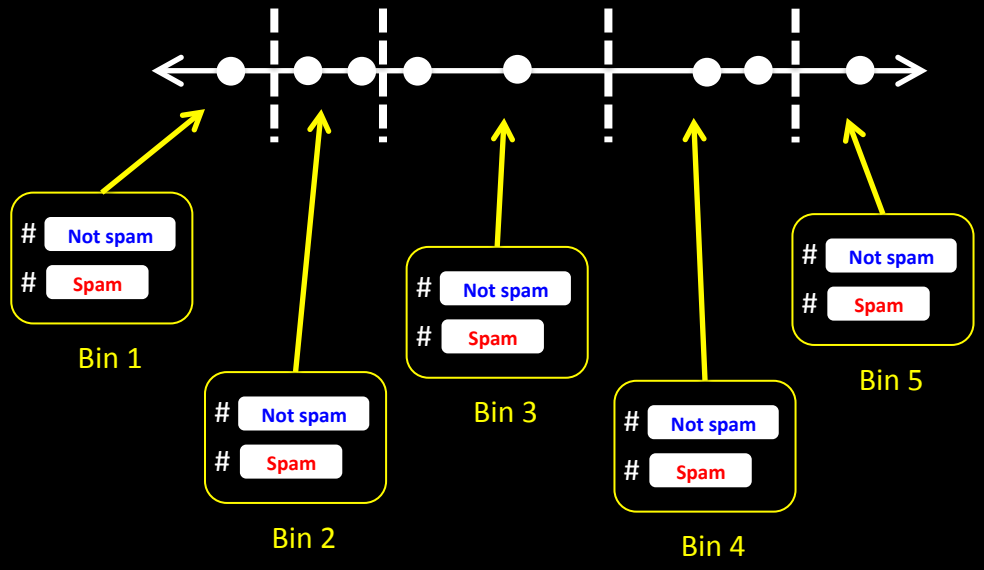
Naively:

possible bins ≈ # instances ☹

Solution: Discretize data

Binning Features

Continuous feature: x_j < (value)



Naively:

possible bins ≈ # instances ☹

Solution: Discretize data

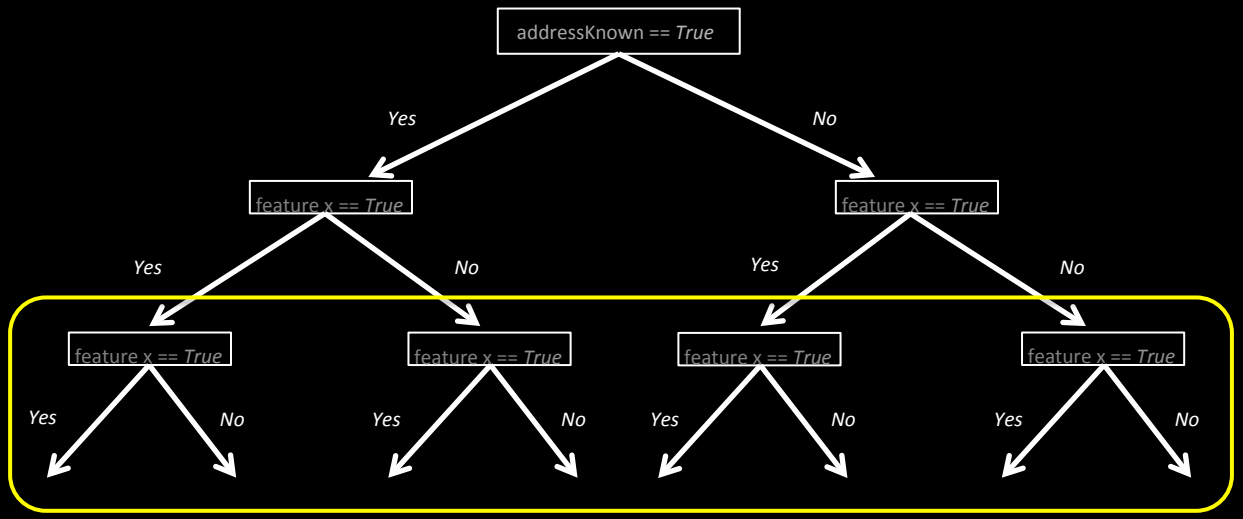
`maxBins`:

larger = higher accuracy

smaller = less communication

databricks

Communication



On each iteration (level)

For each tree node,
For each feature,
For each bin,
Sufficient statistics

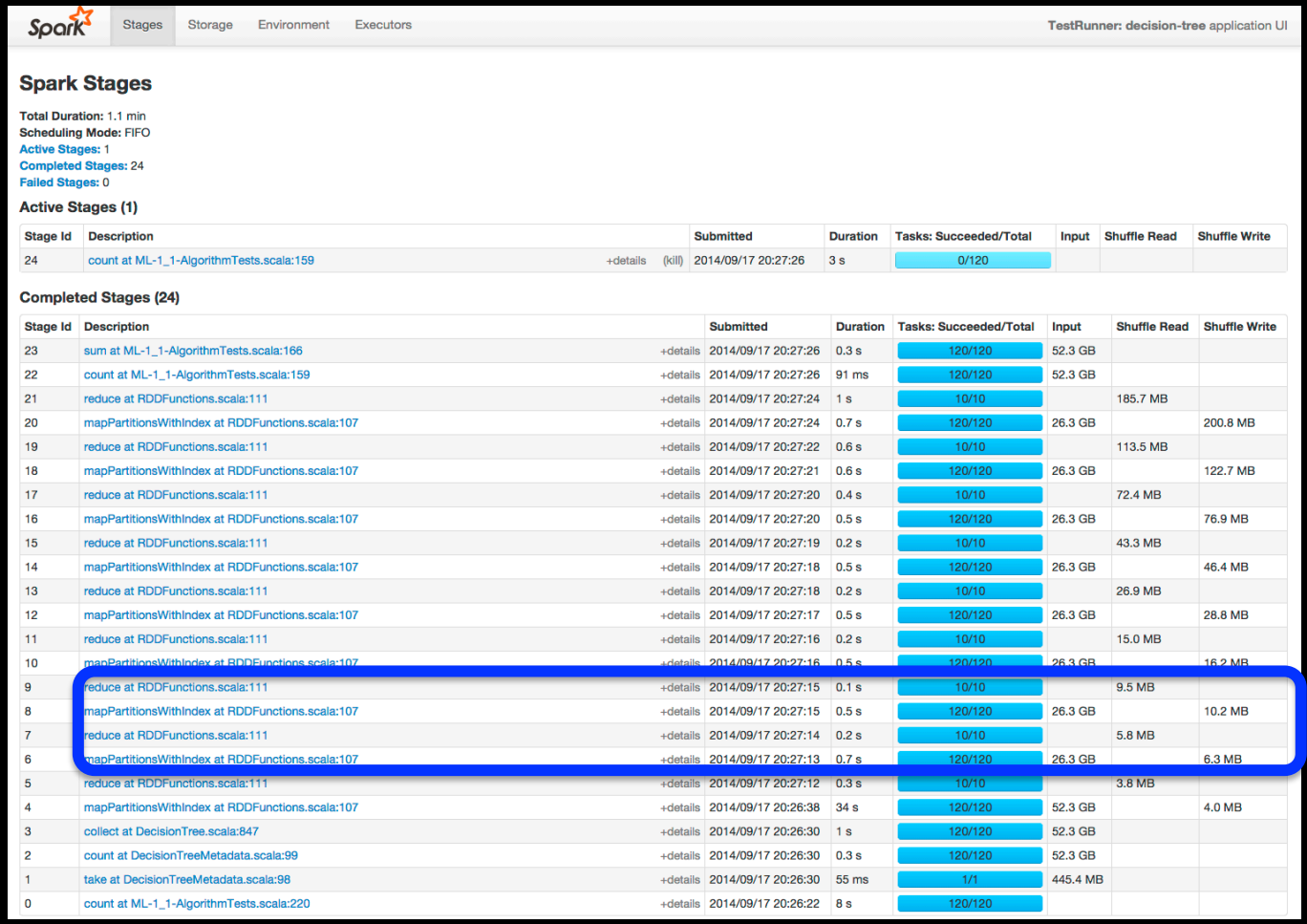
sets of statistics = (# nodes) x (# features) x (# bins/feature)

Set using **`maxBins`** parameter

databricks

Communication

2 million instances, 3500 features, ~70 bins/feature



Spark | Stages | Storage | Environment | Executors | TestRunner: decision-tree application UI

Spark Stages

Total Duration: 1.1 min
Scheduling Mode: FIFO
Active Stages: 1
Completed Stages: 24
Failed Stages: 0

Active Stages (1)

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Shuffle Read	Shuffle Write
24	count at ML-1_1-AlgorithmTests.scala:159 +details (kill)	2014/09/17 20:27:26	3 s	0/120			

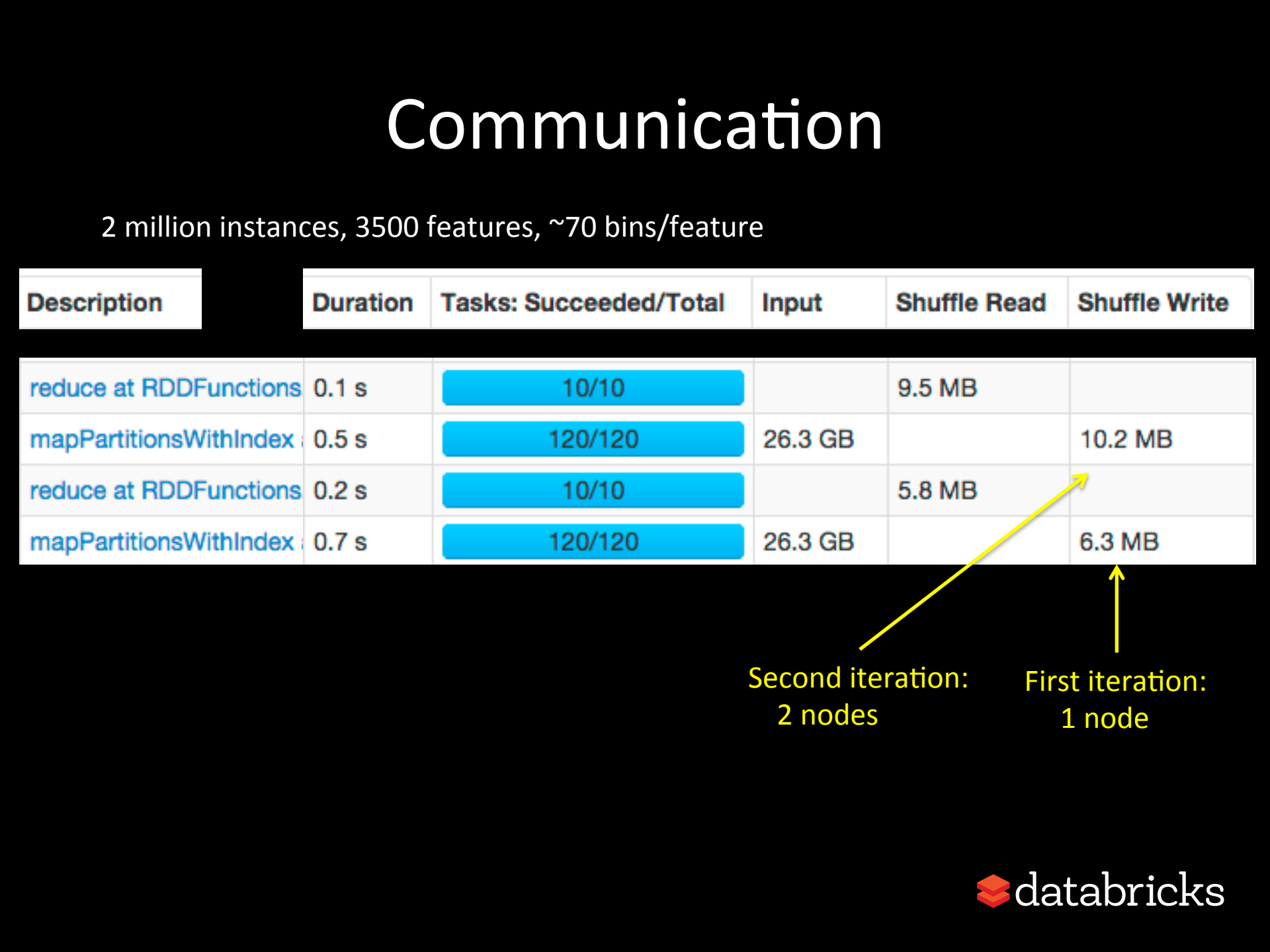
Completed Stages (24)

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Shuffle Read	Shuffle Write
23	sum at ML-1_1-AlgorithmTests.scala:166 +details	2014/09/17 20:27:26	0.3 s	120/120	52.3 GB		
22	count at ML-1_1-AlgorithmTests.scala:159 +details	2014/09/17 20:27:26	91 ms	120/120	52.3 GB		
21	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:24	1 s	10/10		185.7 MB	
20	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:24	0.7 s	120/120	26.3 GB		200.8 MB
19	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:22	0.6 s	10/10		113.5 MB	
18	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:21	0.6 s	120/120	26.3 GB		122.7 MB
17	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:20	0.4 s	10/10		72.4 MB	
16	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:20	0.5 s	120/120	26.3 GB		76.9 MB
15	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:19	0.2 s	10/10		43.3 MB	
14	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:18	0.5 s	120/120	26.3 GB		46.4 MB
13	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:18	0.2 s	10/10		26.9 MB	
12	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:17	0.5 s	120/120	26.3 GB		28.8 MB
11	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:16	0.2 s	10/10		15.0 MB	
10	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:16	0.5 s	120/120	26.3 GB		16.2 MB
9	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:15	0.1 s	10/10		9.5 MB	
8	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:15	0.5 s	120/120	26.3 GB		10.2 MB
7	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:14	0.2 s	10/10		5.8 MB	
6	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:27:13	0.7 s	120/120	26.3 GB		6.3 MB
5	reduce at RDDFunctions.scala:111 +details	2014/09/17 20:27:12	0.3 s	10/10		3.8 MB	
4	mapPartitionsWithIndex at RDDFunctions.scala:107 +details	2014/09/17 20:26:38	34 s	120/120	52.3 GB		4.0 MB
3	collect at DecisionTree.scala:847 +details	2014/09/17 20:26:30	1 s	120/120	52.3 GB		
2	count at DecisionTreeMetadata.scala:99 +details	2014/09/17 20:26:30	0.3 s	120/120	52.3 GB		
1	take at DecisionTreeMetadata.scala:98 +details	2014/09/17 20:26:30	55 ms	1/1	445.4 MB		
0	count at ML-1_1-AlgorithmTests.scala:220 +details	2014/09/17 20:26:22	8 s	120/120			

Communication

2 million instances, 3500 features, ~70 bins/feature

Description	Duration	Tasks: Succeeded/Total	Input	Shuffle Read	Shuffle Write
reduce at RDDFunctions	0.1 s	10/10		9.5 MB	
mapPartitionsWithIndex	0.5 s	120/120	26.3 GB		10.2 MB
reduce at RDDFunctions	0.2 s	10/10		5.8 MB	
mapPartitionsWithIndex	0.7 s	120/120	26.3 GB		6.3 MB



databricks

Communication

2 million instances, 3500 features, ~70 bins/feature

Description	Duration	Tasks: Succeeded/Total	Input	Shuffle Read	Shuffle Write
reduce at RDDFunctions.	1 s	10/10		185.7 MB	
mapPartitionsWithIndex a	0.7 s	120/120	26.3 GB		200.8 MB
reduce at RDDFunctions	0.1 s	10/10		9.5 MB	
mapPartitionsWithIndex a	0.5 s	120/120	26.3 GB		10.2 MB
reduce at RDDFunctions	0.2 s	10/10		5.8 MB	
mapPartitionsWithIndex a	0.7 s	120/120	26.3 GB		6.3 MB

(demo: maxBins)

Setting maxBins parameter

Train trees for maxBins = 2, 4, 8, 16, 32

```
> val treesBins = Array(2, 4, 8, 16, 32).map { maxBins =>
    DecisionTree.trainClassifier(
      mnistTrain,
      numClassesForClassification = 10,
      categoricalFeaturesInfo = Map.empty[Int, Int],
      impurity = "gini",
      maxDepth = 5,
      maxBins = maxBins)
  }

  treesBins: Array[org.apache.spark.mllib.tree.model.DecisionTreeModel] =
  Array(DecisionTreeModel classifier
    If (feature 461 <= 0.0)
     If (feature 378 <= 0.0)
      If (feature 597 <= 0.0)
       If (feature 514 <= 0.0)
        If (feature 623 <= 0.0)
         Predict: 7.0
        Else (feature 623 > 0.0)
         Predict: 0.0
       Else (feature 514 > 0.0)
        If (feature 241 <= 0.0)
         Predict: 6.0
        Else (feature 241 > 0.0)
```

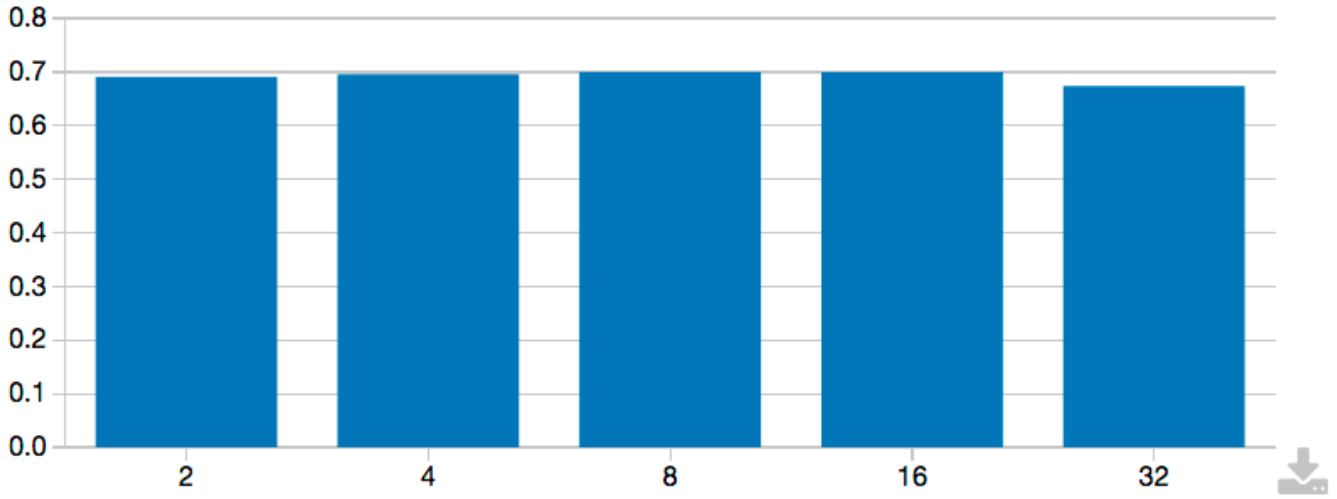
trees – Databricks

https://alpha.databricks.com/#shell/954043

trees (Scala)

Attached: Default Cluster | Run All | Arguments

```
> val binsAccuracies = Array(2, 4, 8, 16, 32).zipWithIndex.map { case
  (maxBins, i) =>
    new Accuracy(maxBins, "gini", accuracy(treesBins(i)))
  }
  display(sc.makeRDD(binsAccuracies))
```



Plot Options...

Command took 2.48s

>

Scala

MLlib Trees in Practice

```
def trainClassifier(
    input: RDD[LabeledPoint],
    numClassesForClassification: Int,
    categoricalFeaturesInfo: Map[Int, Int],
    impurity: String,
    maxDepth: Int,
    maxBins: Int): DecisionTreeModel
```

MLlib Trees in Practice

Good practices:

- maxDepth → Tune with data (model selection)
- maxBins → Set low, increase if needed
- # RDD partitions → Set to # compute cores

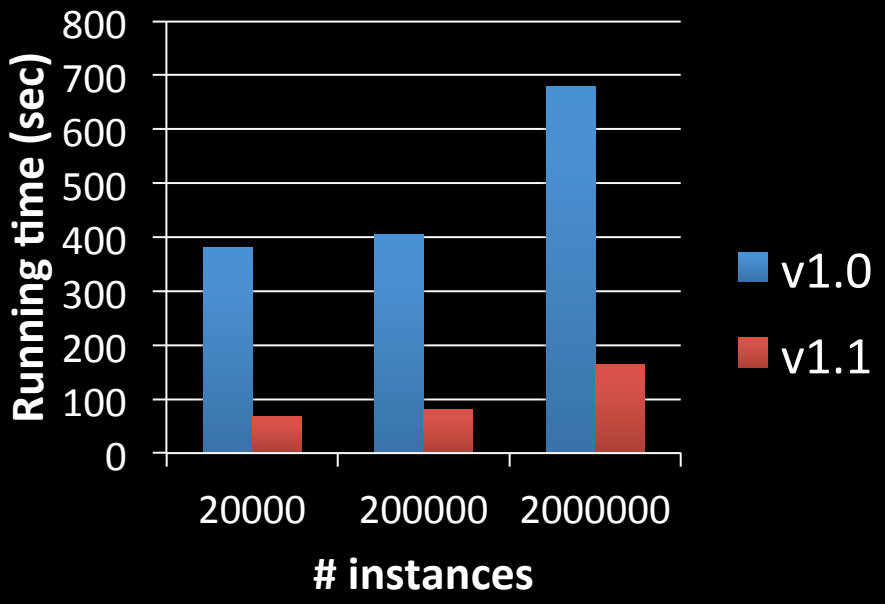
MLlib supports:

- Classification (binary & multiclass labels) & Regression (continuous labels)
- Features: binary, k-category, continuous
- Various impurity measures & other settings
- Python, Scala & Java APIs

Performance Improvements: Spark 1.0 → 1.1

16-node EC2 cluster.
6-level trees.
3500 features.

4-5X faster

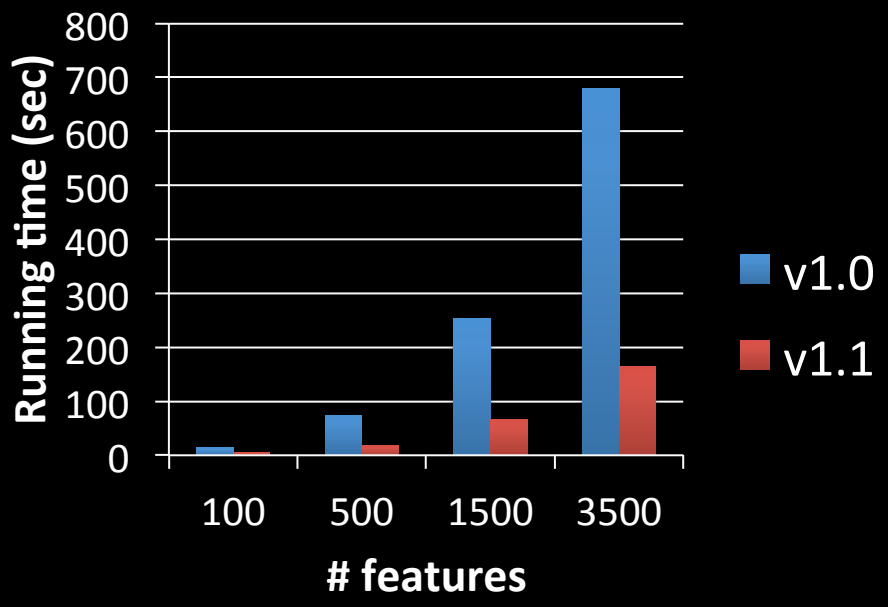


databricks

Performance Improvements: Spark 1.0 → 1.1

16-node EC2 cluster.
6-level trees.
2 million instances.

2-4X faster



databricks

MLlib Trees: Active Development

Ensembles: Random Forests & Boosting
- → PR for random forests
- → Alpine Labs Sequoia Forests: coordinating merge
- → Boosting under development

Model selection pipelines
- → Design doc published on JIRA

More internal optimizations

Where to Go from Here?

- Apache Spark: http://spark.apache.org/
  - Download & try it out
  - Learn with videos, exercises, docs
  - Contribute via Github!
- Databricks: http://databricks.com/
  - Learn about Databricks Cloud!
  - Spark training resources

Summary

- Decision Trees & Spark
- Learning Trees on Spark
- Using MLlib Trees in Practice
  - Model selection
  - Accuracy—communication trade-offs
- Active Development
  - Ensembles (forests & boosting)
  - Model selection
  - More optimizations

Many collaborators
Manish Amde, Hirakendu Das, Evan Sparks, Ameet Talwalkar, Xiangrui Meng, Qiping Li, Sung Chung, Lee Yang, …

Thanks!