Strata+ Hadoop WORLD

PRESENTED BY

O'REILLY cloudera

strataconf.com

#StrataHadoop

Building machine-learning apps with Spark

MLlib, ML Pipelines, and GraphX

Jayant / Amandeep / Krishna / Vartika

Agenda

Overview	10 min (9:00-9:10)	
Lab Environment Setup	15 min (9:10-9:25)	IntelliJ/Scala IDE for Eclipse/Zeppelin
MLlib	90 min (9:25-10:55)	Overview, Linear Regression, Random Forest, Custering, Recommendations, FPG, Text Analytics
Break	10 min (10:55-11:05)	
GraphX	50 min (11:05-11:55)	Overview, Exploring Structures, Community-Affiliation, Algorithms, The AlphaGo Community, Wikepedia Page Rank
ML Pipelines	15 min (11:55-12:10)	CrossValidation
Streaming MLlib	10 min (12:10-12:20)	Streaming K-Means
Closing	10 min (12:20-12:30)	

Download stuff if you haven't yet

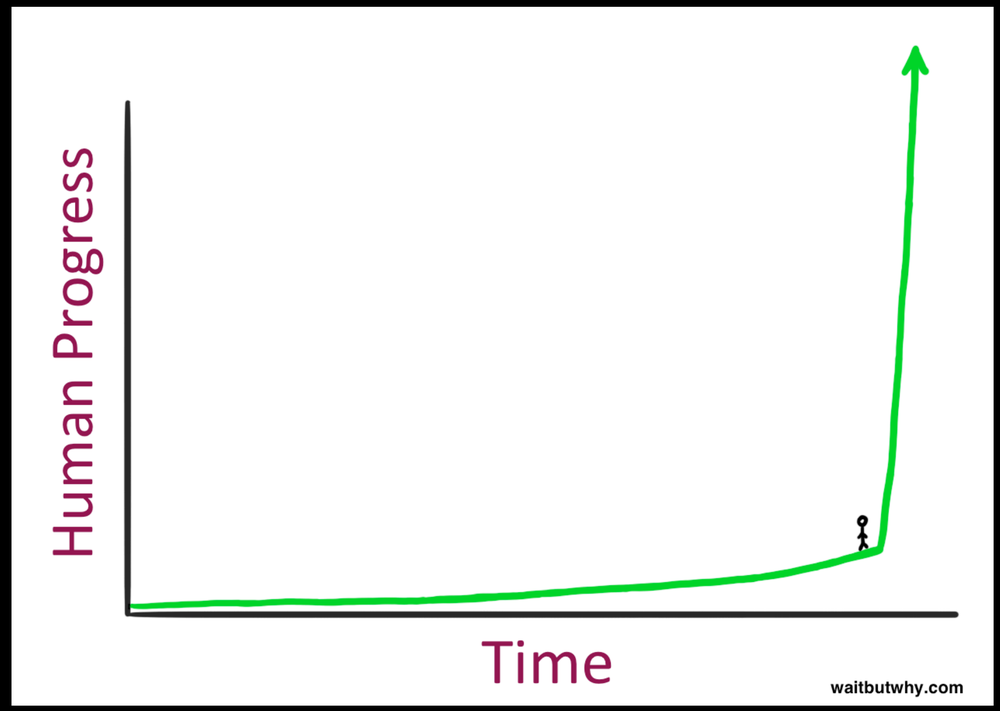
- http://conferences.oreilly.com/strata/hadoop-big-data-ca/public/schedule/detail/46943
- https://github.com/jayantshekhar/strata-2016

Your speakers

- Krishna Shankar
- Jayant Shekar
- Vartika Singh
- Supporting cast: Amandeep Khurana

Why?

- *We are on the edge of change comparable to the rise of human life on Earth. – Vernor Vinge (Prof, SDSU)*
- AI is fast becoming a reality
  - ANI (Narrow)
  - AGI (General)
  - ASI (Super Intelligence)
- We are currently in ANI stage
- To go to AGI and ASI
  - More compute power
  - Better algorithms and systems
- Reference: http://waitbutwhy.com/2015/01/artificial-intelligence-revolution-1.html



Machine Learning & Big Data

- Better systems => ML @ scale
  - Bigger training set => better models => better accuracy
  - Can’t be cost prohibitive
- Spark ecosystem
  - MLlib
  - GraphX
- Others out there
  - H20
  - Dato
  - Graphlab

What's a ML app?

- Collect data
- Clean data – make it usable
- Build model
- Train model
- Test model
- Use model
  - Apply to data at rest (historical)
  - Apply to making decisions as data comes in (current / future)

- MLlib

MLlib

Overview	05 min	
Linear Regression	15 min	Predict House Prices
Random Forest	10 min	Titanic Predict Survival
Clustering	20 min	Topic Modeling on newsgroup data with LDA
Recommendations	10 min	Movie Lens Ratings and Recommendations
FPG	05 min	Shopping Cart Analysis
Text Analytics	25 min	Mood Of the Nation/Mood of the Presidential debates

- Data Types
- Basic Statistics
- Feature Extraction & Transformation

- Local Vector
- Labeled Point
- Local Matrix
- Distributed Matric

- Summary Statistics
- Correlations
- Stratified Sampling
- Hypothesis Testing
- Random Data Generation

- TF-IDF
- Word2Vec
- Tokenizer
- OneHotEncoder
- n-gram

- Classification & Regression
  - Linear Models (SVMs, Linear Regression, Logistic Regression)
  - Naïve Bayes
  - Decision Tree
  - Ensembles
    - Random Forests
    - Gradient Boosted Trees
- Collaborative Filtering
  - ALS
- Frequent Pattern Mining
- Clustering
  - K-Means
- Dimensionality Reduction
  - SVD
  - PCA
- PMML model export

- Titanic Survival Prediction
  - Random Forest

Titanic

Remove the header row when reading

- **Data**

PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked

1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S

2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,71.2833,C85,C

3,1,3,"Heikkinen, Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S

- Target Variable
  - Survived
- Predictor Variables
  - Pclass, Sex, Age, Fare

Titanic DataSet

VARIABLE DESCRIPTIONS:

survival	Survival
	(0 = No; 1 = Yes)
pclass	Passenger Class
	(1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation

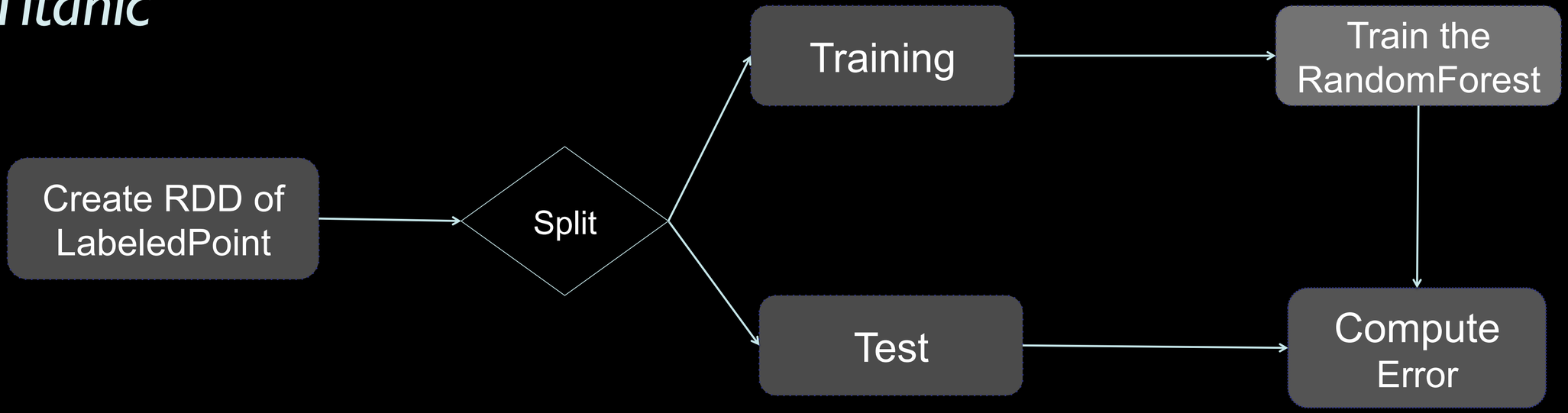
(C = Cherbourg; Q = Queenstown; S = Southampton)

NOTES:

Pclass is a proxy for socio-economic status (SES)
1st ~ Upper; 2nd ~ Middle; 3rd ~ Lower

Age is in Years; Fractional if Age less than One (1)
If the Age is Estimated, it is in the form xx.5

Titanic




```
root
 |-- PassengerId: string (nullable = true)
 |-- Survived: string (nullable = true)
 |-- Pclass: string (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: string (nullable = true)
 |-- SibSp: string (nullable = true)
 |-- Parch: string (nullable = true)
 |-- Ticket: string (nullable = true)
 |-- Fare: string (nullable = true)
 |-- Cabin: string (nullable = true)
 |-- Embarked: string (nullable = true)
```

Random Forest

- **numTrees**: Number of trees in the forest.
- **maxDepth**: Maximum depth of each tree in the forest.
- **categoricalFeaturesInfo**: Specifies which features are categorical and how many categorical values each of those features can take. This is given as a map from feature indices to feature arity (number of categories). Any features not in this map are treated as continuous.
  - E.g., Map(0 -> 2, 4 -> 10) specifies that feature 0 is binary (taking values 0 or 1) and that feature 4 has 10 categories (values {0, 1, ..., 9}). Feature indices are 0-based: features 0 and 4 are the 1st and 5th elements of an instance's feature vector.

- Tree 0:
- If (feature 0 in {0.0})
- If (feature 4 <= 8.7125)
- If (feature 3 <= 0.0)
- If (feature 2 <= 0.0)
- Predict: 0.0
- Else (feature 2 > 0.0)
- Predict: 0.0
- Else (feature 3 > 0.0)
- If (feature 1 <= 0.42)
- Predict: 1.0
- Else (feature 1 > 0.42)
- Predict: 0.0
- Else (feature 4 > 8.7125)
- If (feature 1 <= 14.0)
- If (feature 2 <= 2.0)
- Predict: 1.0
- Else (feature 2 > 2.0)
- Predict: 0.0
- Else (feature 1 > 14.0)

- Tree 1:
- If (feature 0 in {0.0})
- If (feature 4 <= 9.8375)
- If (feature 4 <= 7.8958)
- If (feature 4 <= 7.8292)
- Predict: 0.0
- Else (feature 4 > 7.8292)
- Predict: 0.0
- Else (feature 4 > 7.8958)
- If (feature 2 <= 0.0)
- Predict: 0.0
- Else (feature 2 > 0.0)
- Predict: 1.0
- Else (feature 4 > 9.8375)
- If (feature 3 <= 0.0)
- If (feature 4 <= 26.0)
- Predict: 0.0
- Else (feature 4 > 26.0)
- Predict: 0.0
- Else (feature 3 > 0.0)

- Recommendations
  - Movie Lens Ratings

MovieLens 100K Dataset

Stable benchmark dataset. 100,000 ratings from 1000 users on 1700 movies. Released 4/1998.

- README.txt
- ml-100k.zip (size: 5 MB, checksum)
- Index of unzipped files

Permalink: http://grouplens.org/datasets/movielens/100k/

MovieLens 1M Dataset

Stable benchmark dataset. 1 million ratings from 6000 users on 4000 movies. Released 2/2003.

- README.txt
- ml-1m.zip (size: 6 MB, checksum)

Permalink: http://grouplens.org/datasets/movielens/1m/

MovieLens 10M Dataset

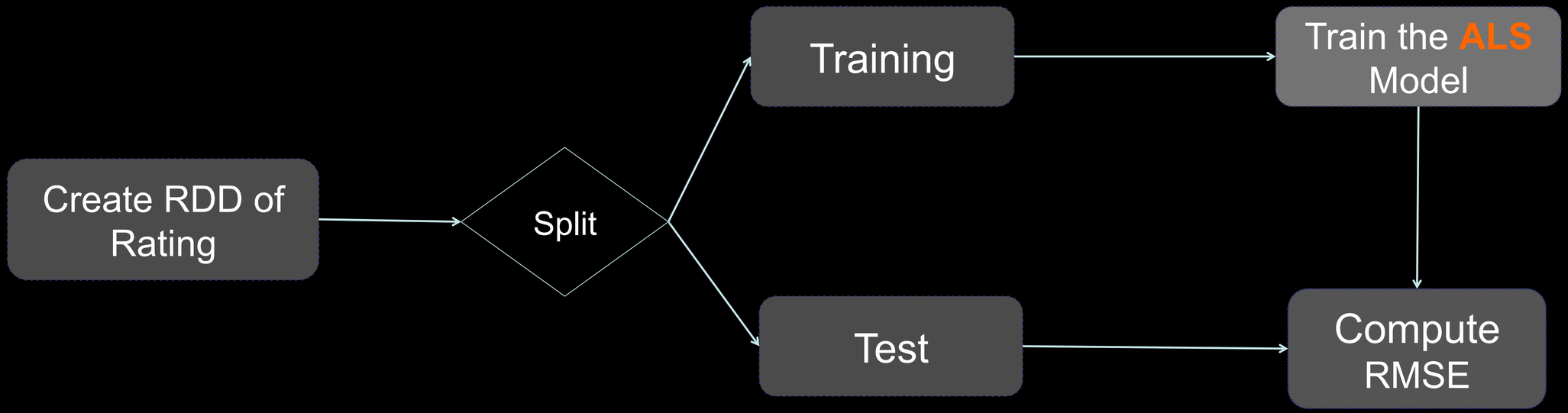
Stable benchmark dataset. 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009.

- README.html
- ml-10m.zip (size: 63 MB, checksum)

MovieLens

Userid, movie id, rating

0::2::3
0::3::1
0::5::2



Recommendations with ALS

- Fill in the missing entries of a user-item association matrix
- *numBlocks* is the number of blocks used to parallelize computation (set to -1 to auto-configure).
- *rank* is the number of latent factors in the model.
- *iterations* is the number of iterations to run.
- *lambda* specifies the regularization parameter in ALS.
- *implicitPrefs* specifies whether to use the *explicit feedback* ALS variant or one adapted for *implicit feedback* data.
- *alpha* is a parameter applicable to the implicit feedback variant of ALS that governs the *baseline* confidence in preference observations.

discover actual shopping behavior

- Frequent Pattern Mining
  - FPG

Frequent Pattern Mining

- Mllib has parallel implementation of FP-Growth

  - minSupport: the minimum support for an itemset to be identified as frequent. For example, if an item appears 3 out of 5 transactions, it has a support of 3/5=0.6.
  - numPartitions: the number of partitions used to distribute the work.

FPGrowth

Create RDD of ArrayList<String>

r z h k p
z y x w v
u t s
s x o n r
x z y m t
s q e

Run FPGrowth

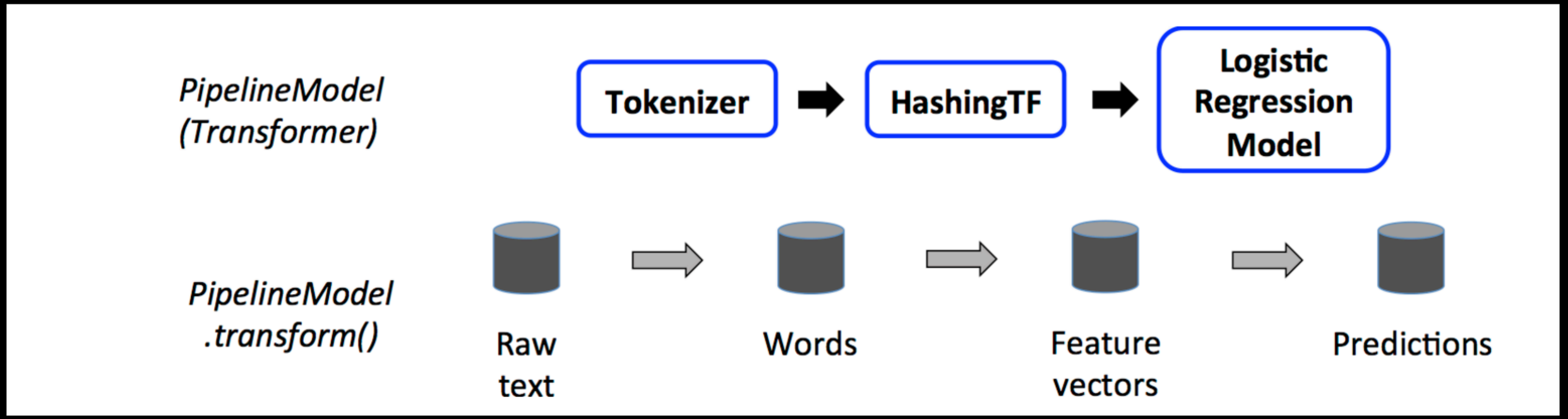
[s], 3
[s,x], 3
[s,x,z], 2
[s,z], 2
[r], 3
[r,x], 2
[r,z], 2
[y], 3
[y,s], 2
[y,s,x], 2

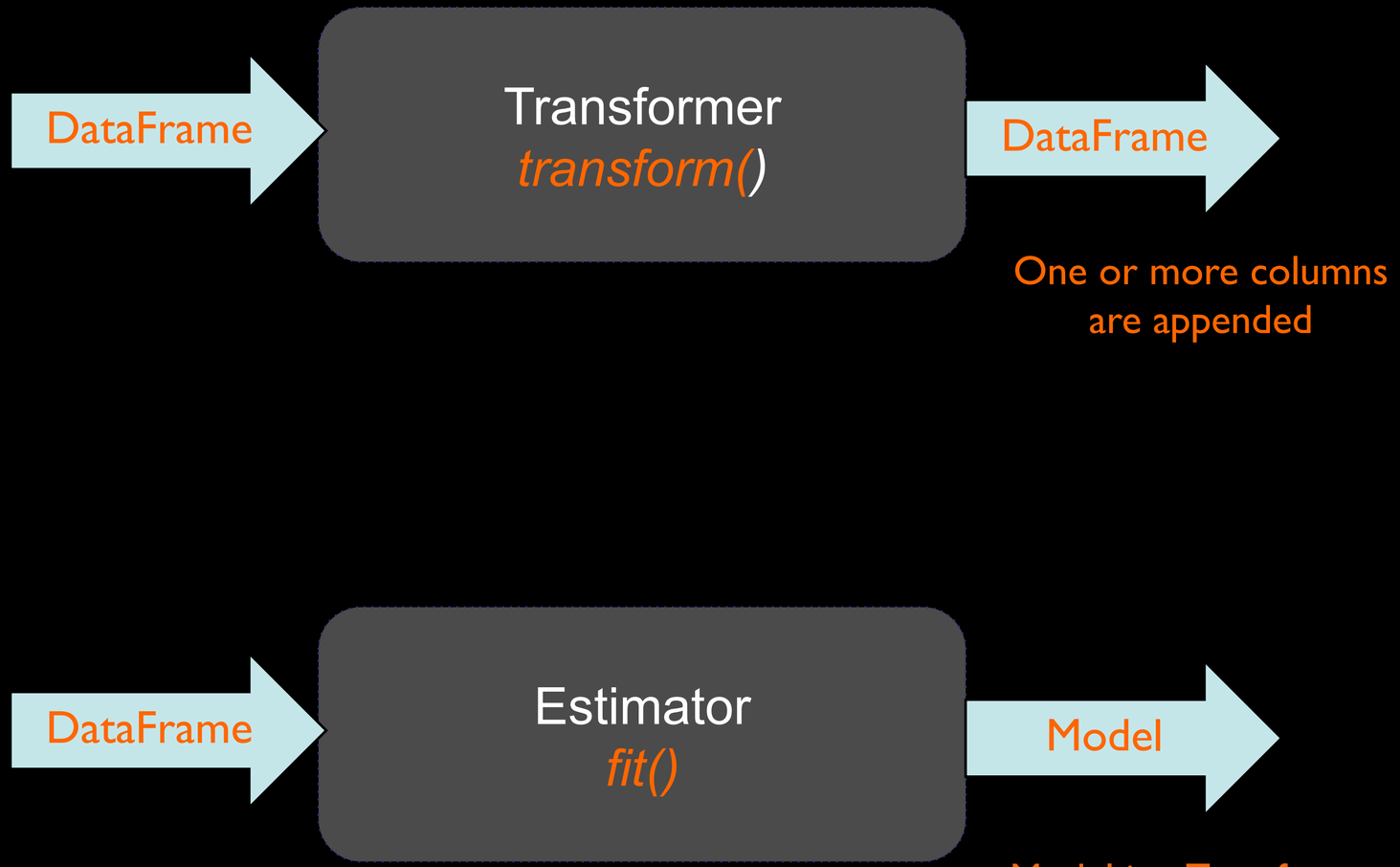
Print Results

- ML Pipelines
- 15 min

Spark ML

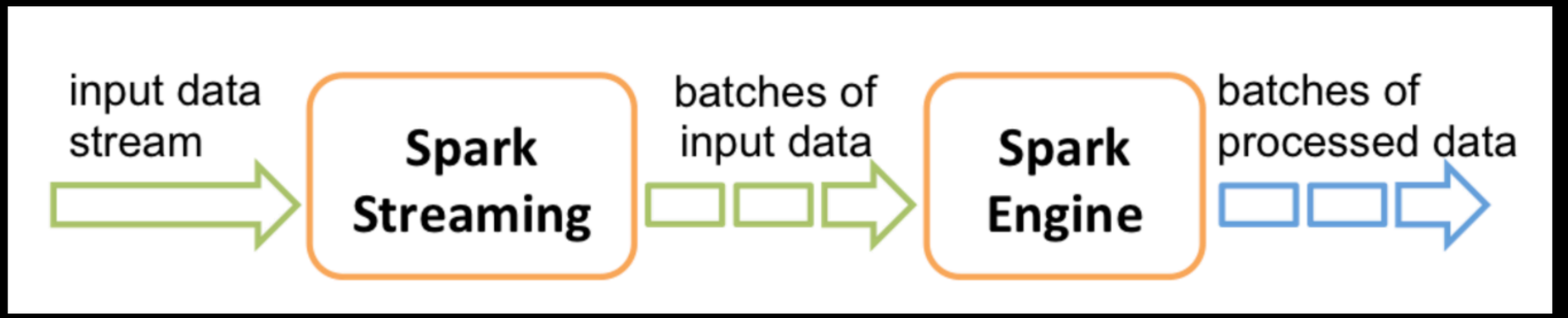
- DataFrames
- Transformer
- Estimator
- Pipeline



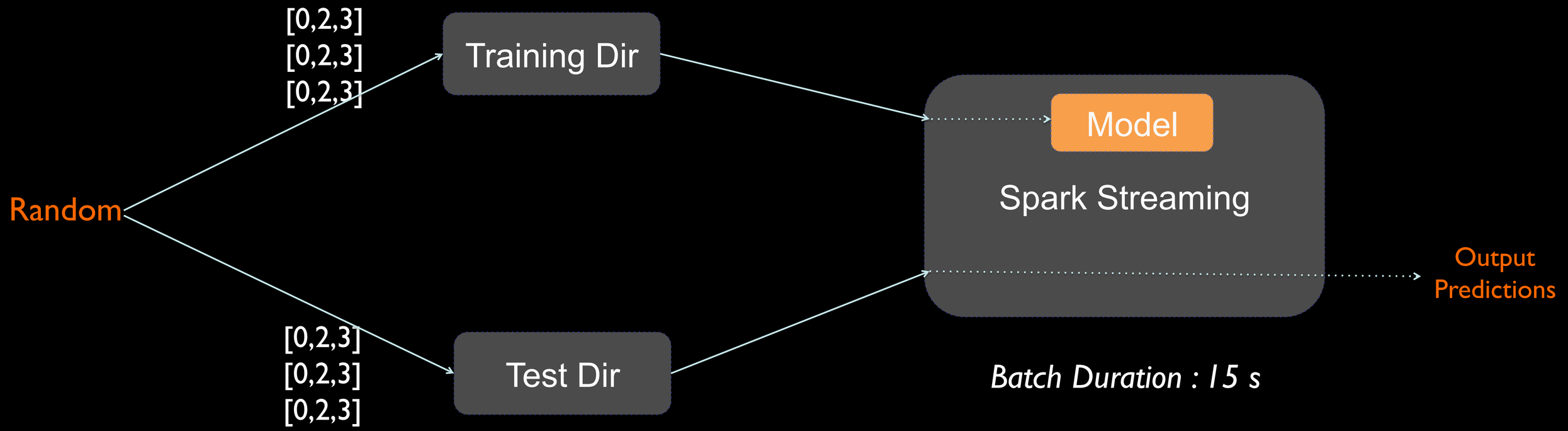

DataFrame
Transformer
transform()
DataFrame
One or more columns
are appended
DataFrame
Estimator
fit()
Model
Model is a Transformer

- Streaming MLlib
- 10 min


Kafka
Flume
HDFS/S3
Kinesis
Twitter
Spark Streaming
HDFS
Databases
Dashboards


input data stream
Spark Streaming
batches of input data
Spark Engine
batches of processed data

Streaming K-Means



Estimate clusters on one stream of data and make predictions on another stream

Streaming K-Means

- Each training point should be formatted as **[x1, x2, x3]**
- Test data point should be formatted as **(y, [x1, x2, x3])**, where y is some useful label or identifier (e.g. a true category assignment).
- Anytime a text file is placed in **../trainingDir** the model will update
- Anytime a text file is placed in **../testDir** they would be processed to produce predictions using the current model
- The decay can be specified using a halfLife parameter, which determines the correct decay factor such that, for data acquired at time t, its contribution by time t + halfLife will have dropped to 0.5