BIG DATA

Machine learning Case Study

Spark MLLib

K V Subramaniam

Computer Science and Engineering

Motivational Problem: Text Classification

Goal : Given a text Document , Predict its topic.

Dataset: "20 Newsgroups"
From UCI KDD Archive

Features

```
Subject: Re: Lexan Polish?
Suggest McQuires #1 plastic
polish.  It will help somewhat
but nothing will remove deep
scratches without making it
worse than it already is.
McQuires will do something...
```

text, image, vector, ...

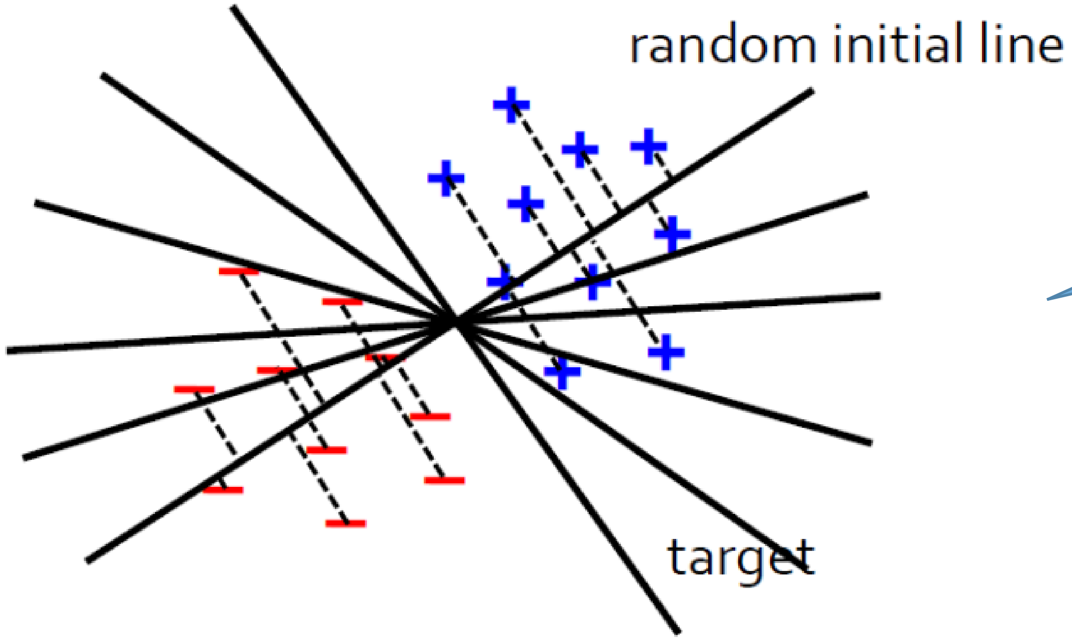
⟹

Label

1: about science
0: not about science

CTR, inches of rainfall, ...

Goal : Find best line separating two sets of points.



Classify into science and non-science. Each point represents a document.

[Further details on logistic regression can be found at - https://en.wikipedia.org/wiki/Logistic_regression](https://en.wikipedia.org/wiki/Logistic_regression)

Machine Learning Workflow and Challenges

Example ML Workflow

TRAINING	TESTING/PRODUCTION
Load data	Load ***new*** data
↓ *labels + plain text*	↓ *plain text*
Extract features	Extract features
↓ *labels + feature vectors*	↓ *feature vectors*
Train model	Predict using model
↓ *labels + predictions*	↓ *predictions*
Evaluate	Act on predictions

Almost identical workflow

What are the pain points?

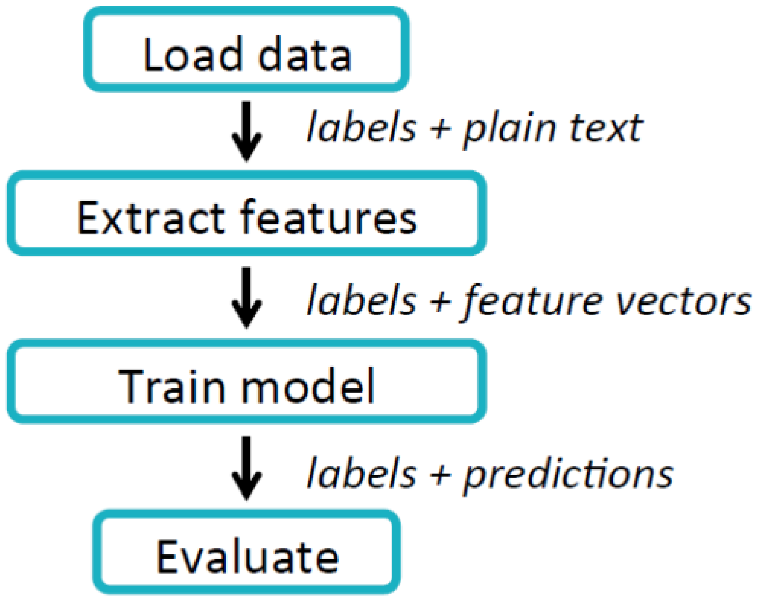
- Create and Handle Many RDDs and data types
  - Labels, features, predictions..
- Write as a script
  - Whole pipeline needs to be coded as a script
  - Not modular
- Tune parameters
  - Key part of ML
  - Training many models
    - For different splits of data
    - Different sets of parameters

Solving the Machine learning challenges

How can the task be made easier?

- Make RDDs easier to read
  - Have to explicitly break up the fields in RDD
  - E.g., break line into blank separated tokens
- As developers we would like to just
  - Program to extract features
  - Specify the model to be used
- However, ML needs additional work
  - Write a script to do all the steps
  - Train the model
  - Evaluate the error of the model by testing

TRAINING



Key concepts

- Reading RDDs: DataFrame

  Solves the RDD creation pain-point

- ML Pipeline
  - Transformers
  - Estimators
  - Evaluators

  Solves the Scripting..

- Parameters
  - API
  - Tuning

  Solves the parameter tuning pain point

Dataframes

- Recall
- Announced Feb 2015
- Inspired by data frames in R and Pandas in Python
- Works in:

http://training.databricks.com/intro.pdf

What is a Dataframe?

- a distributed collection of data organized into named columns
- Like a table in a relational database

Features

- Scales from KBs to PBs
- Supports wide array of data formats and storage systems (Hive, existing RDDs, etc)
- State-of-the-art optimization and code generation via Spark SQL Catalyst optimizer
- APIs in Python, Java

Dataframe : RDD + Schema + DSL

Named columns with types

```
label: Double
text: String
words: Seq[String]
features: Vector
prediction: Double
```

label	text	words	features
0	This is ...	["This", "is", ...]	[0.5, 1.2, ...]
0	When we ...	["When", ...]	[1.9, -0.8, ...]
1	Knuth was ...	["Knuth", ...]	[0.0, 8.7, ...]
0	Or you ...	["Or", "you", ...]	[0.1, -0.6, ...]

Domain-Specific Language

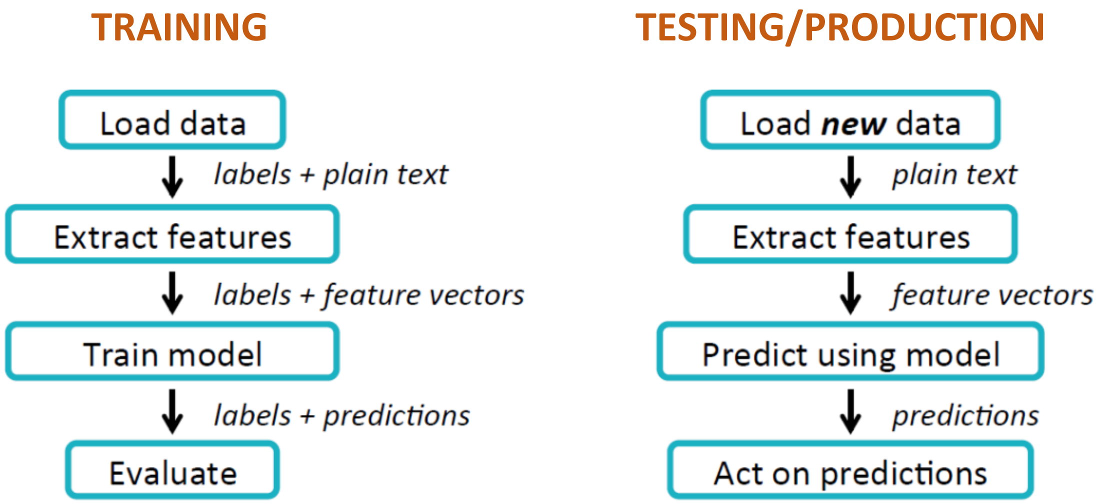
```
# Select science articles
sciDocs =
  data.filter("label" == 1)

# Scale labels
data("label") * 0.5
```

ML Pipelines

ML Pipelines

- Introduced in Spark 1.2 and 1.3
- Allows developers to just
  - Program to extract features
  - Specify the model to be used
- Automates the process of
  - Write a script to do all the steps
  - Train the model
  - Evaluate the error of the model by testing
  - Or deploy in production



The ML Pipeline

Transformers

- Extract features from DataFrame
- Features are stored in a new DataFrame

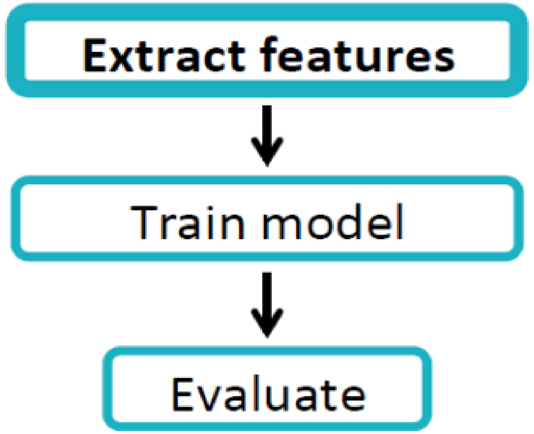
Estimators

- ML Algorithms
- MLLib has standard defined ML algorithms (e.g., Logistic Regression)
- User can add his own

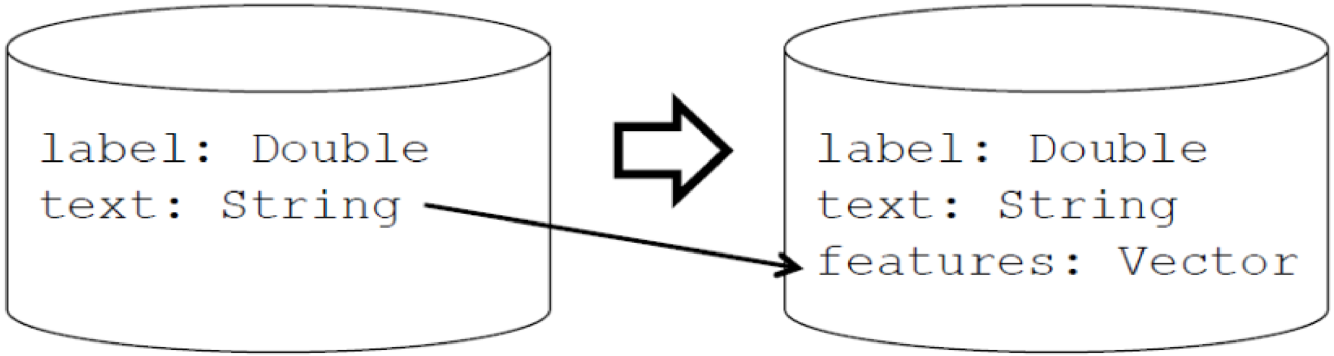
Evaluators

- Compute predictions and estimate metrics such as error
- Tune algorithm parameters
- Evaluator depends upon estimator
  - Evaluator that trains Logistic Regression cannot be used for Decision Trees

TRAINING




```
def transform(DataFrame): DataFrame
```



Label	Text
0	
1	

Label	Text	Features
0		
1		

Estimator

TRAINING

Extract features

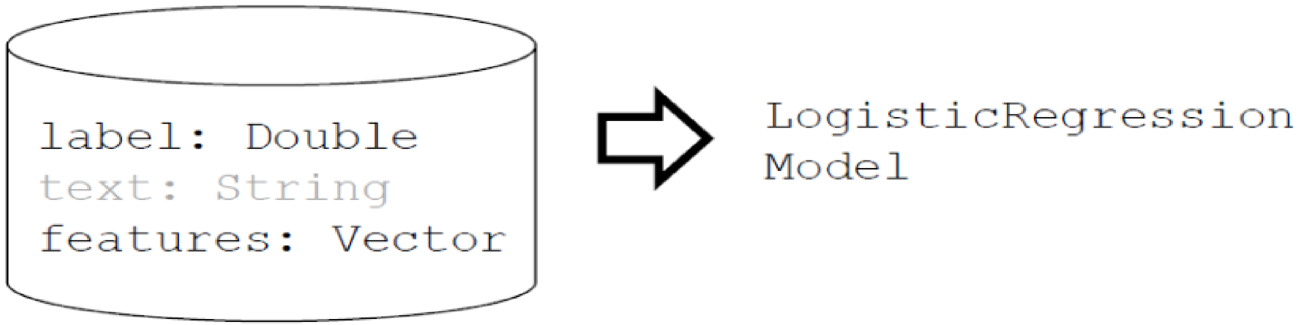
↓

Train model

↓

Evaluate

`def fit(DataFrame): Model`



Label	Text	Features
0		
1		

Evaluator

TRAINING

Extract features

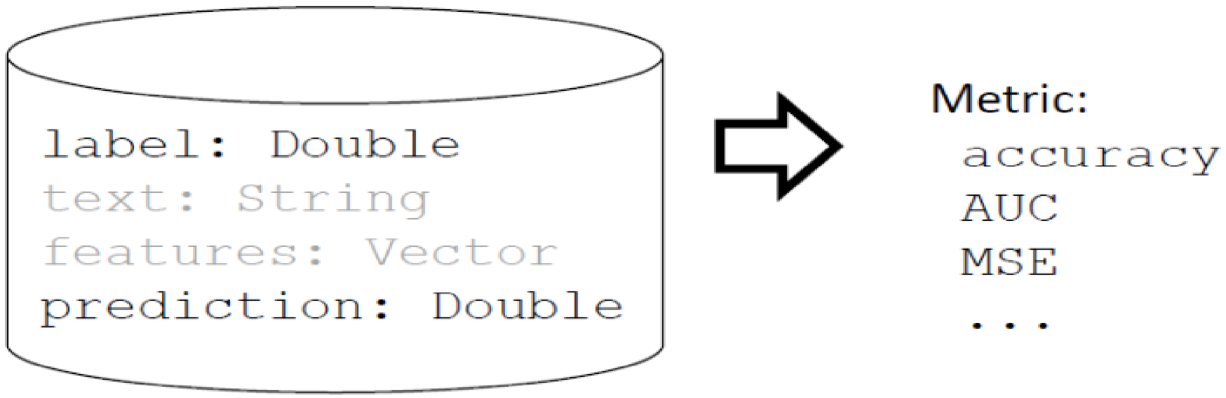
↓

Train model

↓

Evaluate

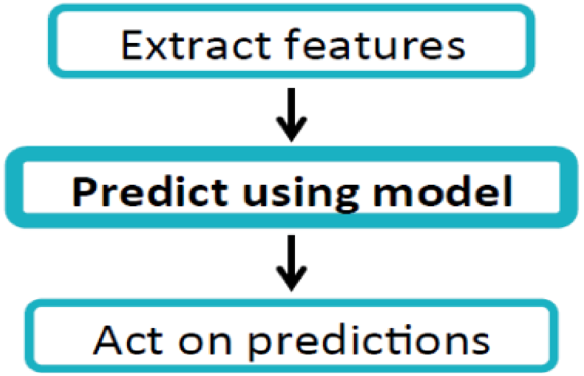
```
def evaluate(DataFrame): Double
```



Label	Text	Features	Prediction
0			
1			

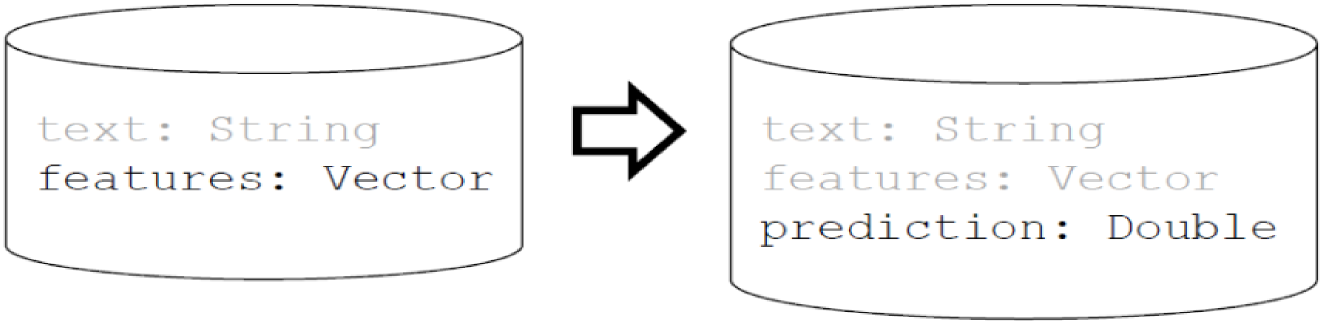
Model

TESTING/PRODUCTION



Model is a type of Transformer

```
def transform(DataFrame): DataFrame
```

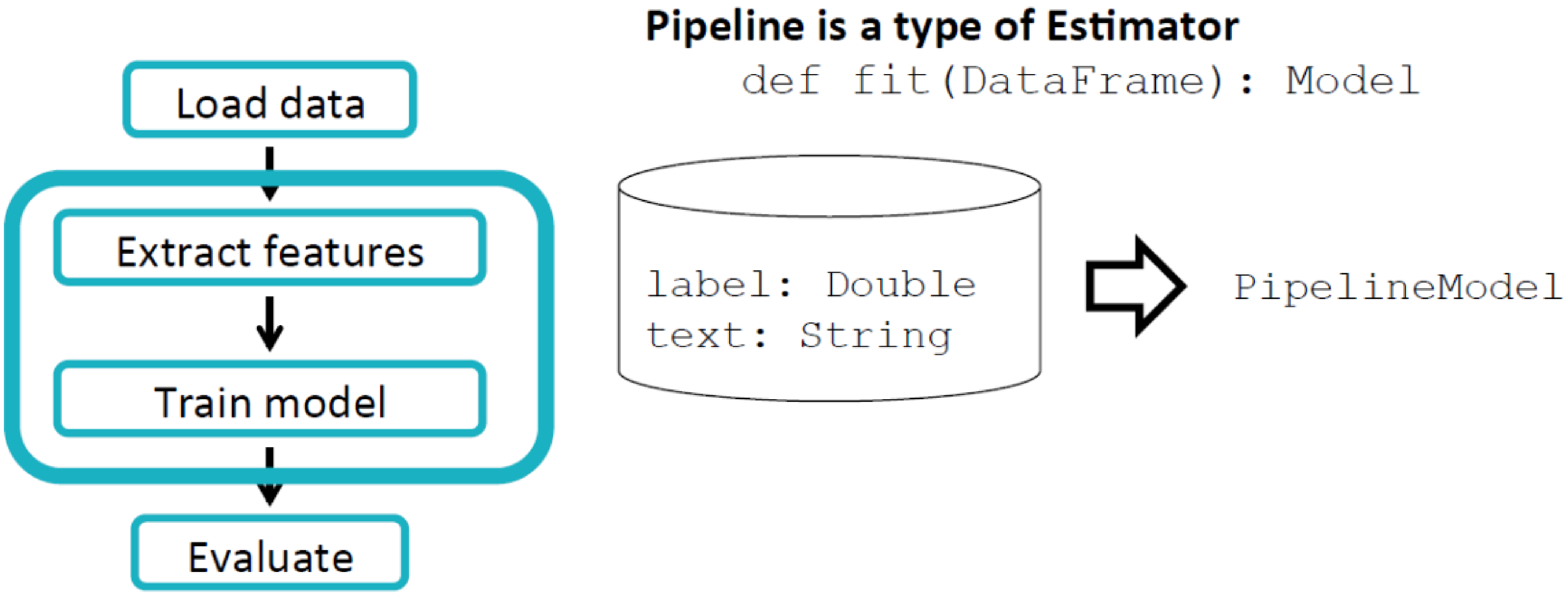


Text	Features

Text	Features	Prediction

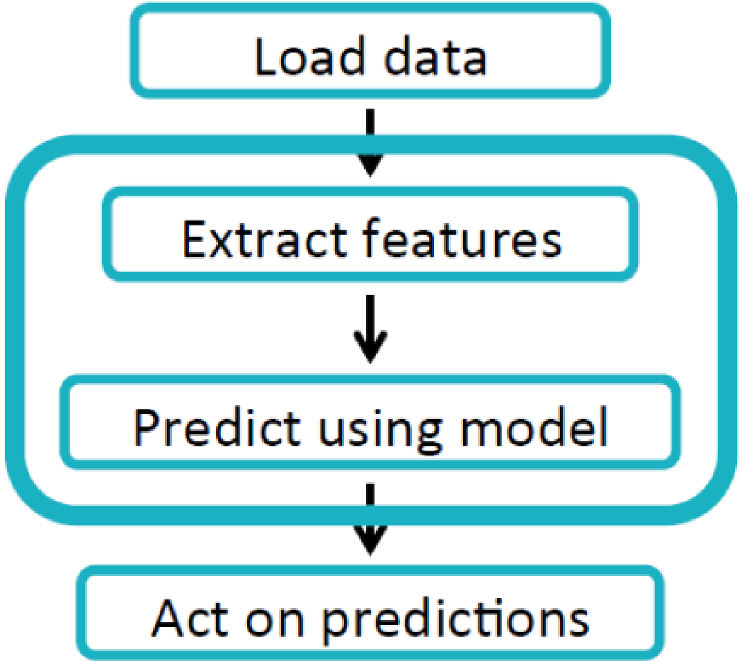
The training Pipeline

TRAINING



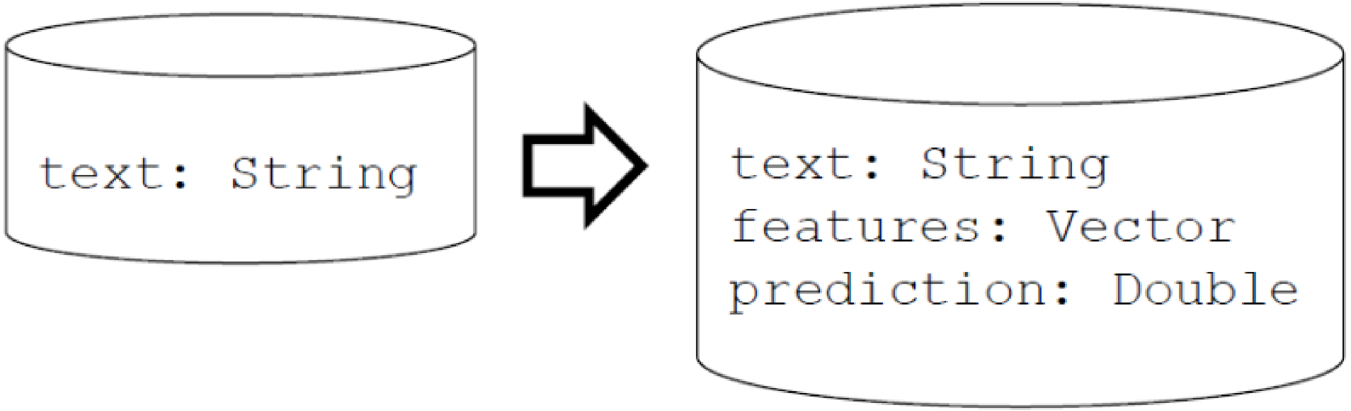
The testing pipeline/model

TESTING/PRODUCTION

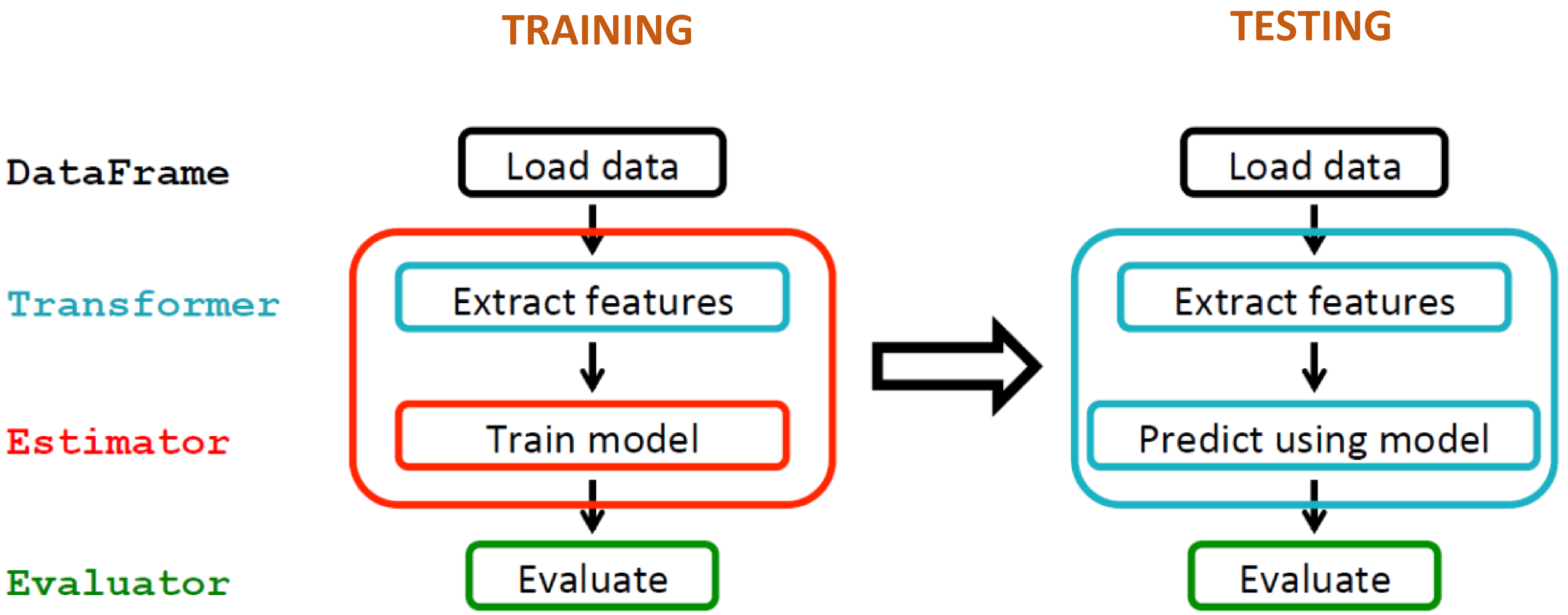


PipelineModel is a type of Transformer

```
def transform(DataFrame): DataFrame
```



Putting it all together

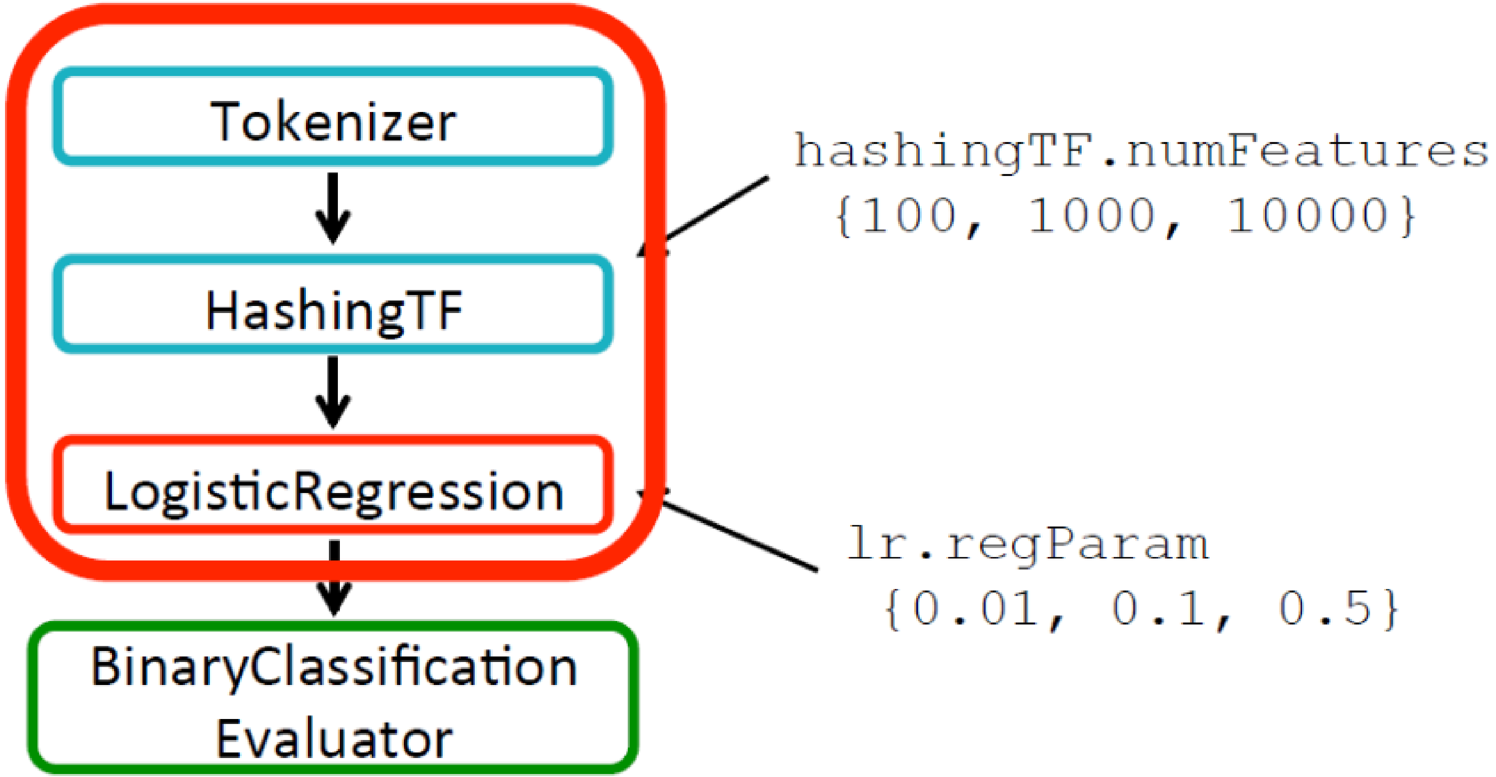


Parameter Tuning

Given:
- Estimator
- Parameter Grid
- Evaluator

Find the Best parameters

CrossValidator



Exercise 4: 10 minutes

- Suppose we have a dataset in which each line has a recording of a noise, and its classification
- E.g., <bell.wav>, bell

- What would be the input DataFrame be?
- Suppose we want to recognize sounds by
  - Extracting the frequencies from the wav file
  - Gaussian model
    - Find the average frequency of each sound
    - For a new sound, calculate average frequency
    - Find closest matching sound
- What are the DataFrames, Evaluators, etc needed.

Exercise 4: Solution

- Suppose we have a dataset in which each line has a recording of a noise, and its classification
- E.g., <bell.wav>, bell

- Input DataFrame
  - <bell.wav>, bell
- Feature DataFrame
  - <bell.wav>, bell, frequencies
- Transformer (use same transformer for train/ predict)
  - <bell.wav> Bell, average frequency
- Model
  - train(FeatureDataFrame)
    - Associate average frequency for "Bell"
  - predict(PredictDataFrame)
    - Output closest matching sound

Mllib algorithms

- Classification
  - Logistic Regression
  - Decision Tree
  - Random Forest
  - Gradient boosted tree
  - Multilayer Perceptron
  - SVM
  - Naïve Bayes
- Clustering
  - K-Means
  - LDA
  - GMM
- Collaborative Filtering
  - ALS
- Frequent Pattern Mining

Deep Learning with Big Data

Challenges for Deep Learning

- Heterogenous cluster
- Deep Learning (Tensorflow)
  - Iterative
  - Matrix vector multiplication – Linear algebra
- Initially evolved on a single machine – only scale up
- Then had its own cluster
  - Typically heterogenous with CPUs, GPUs, TPUs

Challenges for Deep Learning

- But data resides on HDFS and big data platform uses Spark
- How should the two work together.
- Typically the two clusters are different

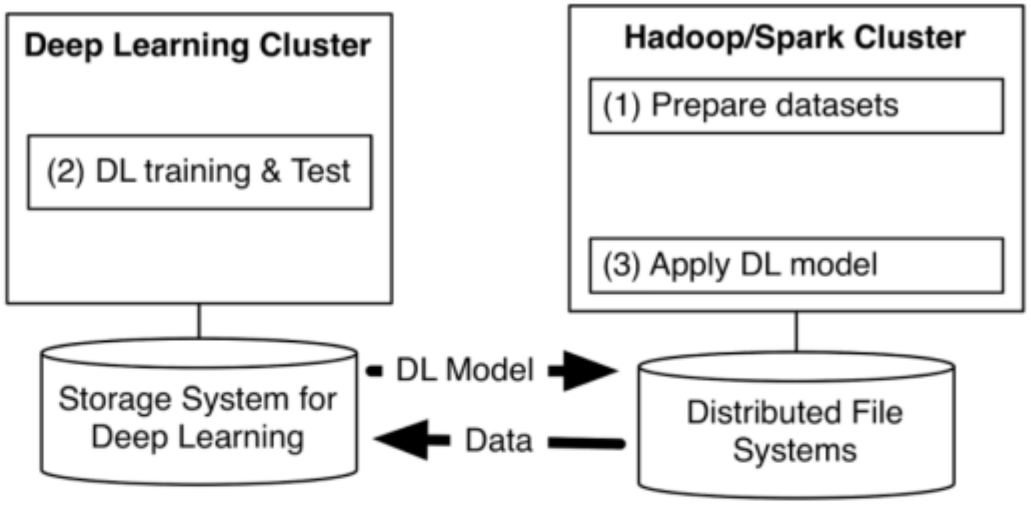


Figure 1: ML Pipeline with multiple programs on separated clusters

https://developer.yahoo.com/blogs/157196317141/

Challenges for Deep Learning

- Can we use the same cluster?

- Tensorflow on Spark
  - From Yahoo

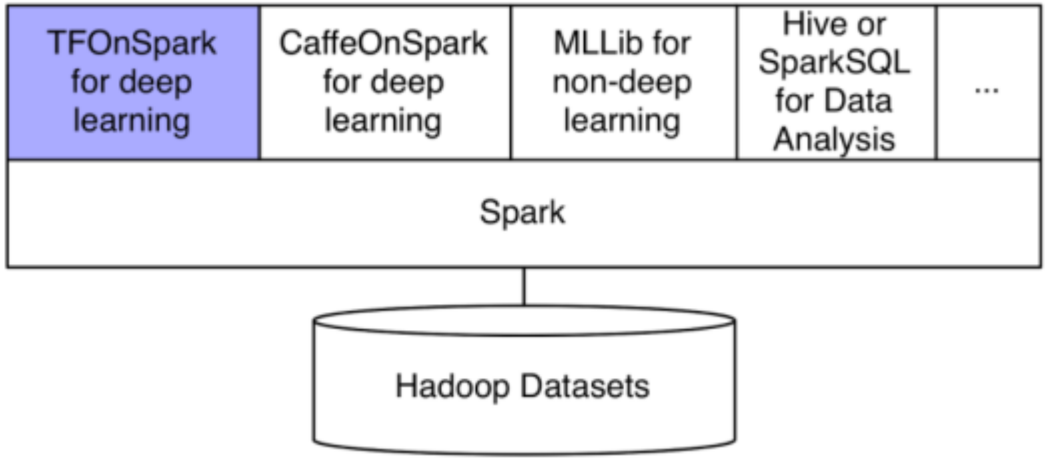


Figure 2: TensorFlowOnSpark for deep learning on Spark clusters

https://developer.yahoo.com/blogs/157196317141/

BIG DATA

Tensorflow on Spark Architecture

- Supports both
  - Model parallelism
  - Data parallelism
- <10 lines of code change reqd
- Algorithm and parameter server rui on Spark executors
  - Can read data directly from HDFS
  - Spark RDD data is fed to spark executor which passes it to Tensorflow
- RDMA → faster network transfers

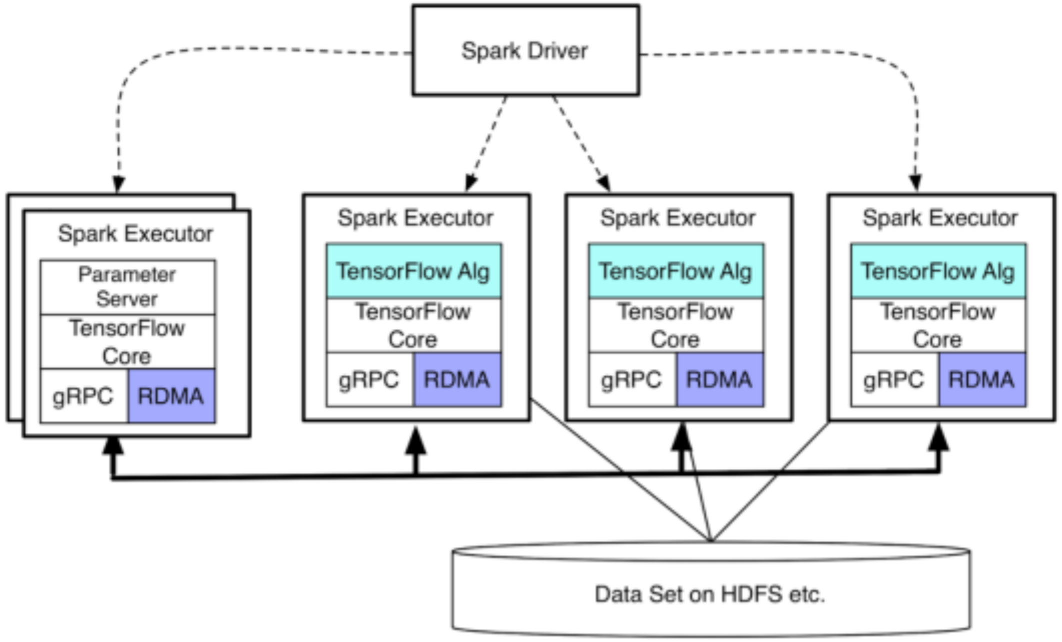


Figure 3: TensorFlowOnSpark system architecture

https://developer.yahoo.com/blogs/157196317141/

Other solutions

- SystemML – systemml.apache.org
  - IBM
  - SystemML: Declarative Machine Learning on Spark http://www.vldb.org/pvldb/vol9/p1425-boehm.pdf
  - Uses a declarative ML language
  - Translated to MR/Spark
- Intel BigDL
  - Modeled on Torch

THANK YOU

K V Subramaniam, Usha Devi
Dept. of Computer Science and Engineering

subramaniamkv@pes.edu
ushadevibg@pes.edu