Apache Spark

KEEPCODING
Tech School

Antonio Gutiérrez López
gutierrez.lopez.ant@gmail.com

Spark MLlib

KEEPCODING
Tech School

Antonio Gutiérrez López
gutierrez.lopez.ant@gmail.com

Spark MLlib

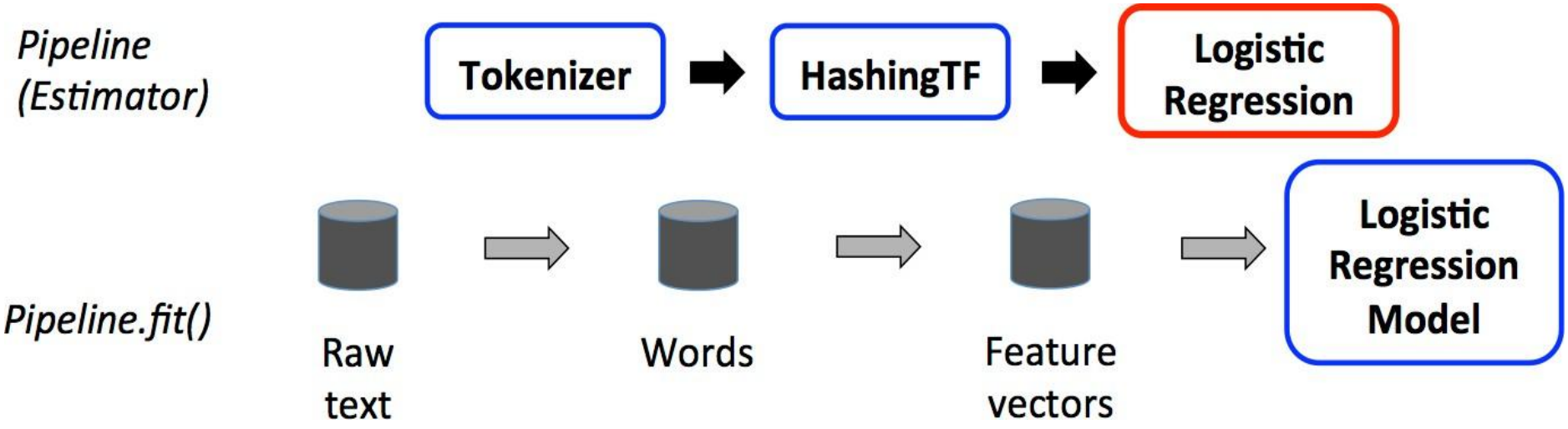
- Spark MLlib es una librería de spark que nos permite hacer jobs de machine learning en modo distribuido.
- Proporciona:
  - **ML Algortihms:** clasificación, regresión, clusterizado y filtros colaborativos.
  - **Featurization:** extracción de features, transformaciones, reducción de dimensiones.
  - **Pipelines:** herramientas para construir, evaluar y tunear modelos de ML.
  - **Persistence:** salvar y guardar modelos y pipelines.
  - **Utilidades:** trabajar con datos, estadísticas, algebra lineal, etc

Spark MLlib funciona utilizando el API de **DataFrame**.

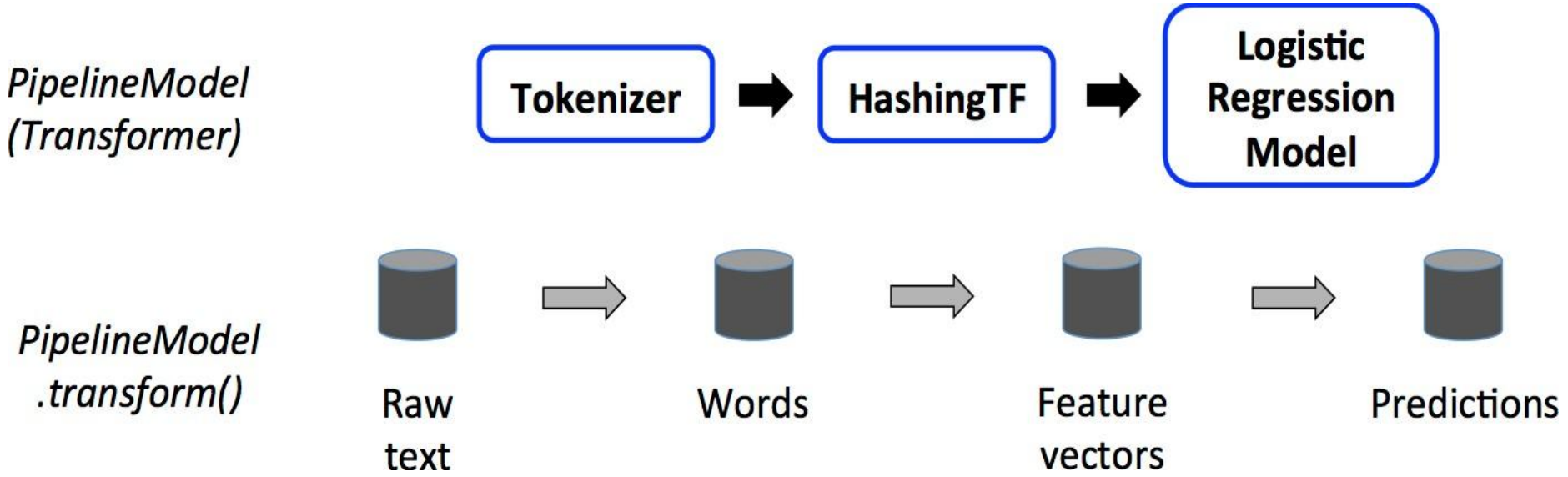
Pipelines

- Los pipelines son flujos de trabajo que combinan múltiples algoritmos y operaciones.
- Conceptos:
  - **DataFrame:** La API de ML usa DataFrame de SparkSQL, en estos dataframe se preparan los datos para los modelos y se obtienen los resultados.
  - **Transformer:** Transformar un DataFrame en otro DataFrame, por ejemplo transformar un dataFrame de features a uno de predicciones.
  - **Estimator:** Algoritmos que se puede entrenar y produce un transformer.
  - **Pipeline:** Conjunto encadenado de *transformers* y *estimators*.
  - **Parameter:** Parámetros asociados a los *transformers* y *estimators*

Pipelines: Train Time



Pipelines: Test Time



Algorithms

- Classification & Regression

https://spark.apache.org/docs/latest/ml-classification-regression.html

- Clustering

https://spark.apache.org/docs/latest/ml-clustering.html

- Collaborative Filtering

https://spark.apache.org/docs/latest/ml-collaborative-filtering.html

- Frequent Pattern Mining

https://spark.apache.org/docs/latest/ml-frequent-pattern-mining.html

Tunning

- Spark MLlib nos permite tunear los algoritmos realizando pruebas en base a parámetros de entrada.
    - Indicamos que parámetros de entrada queremos probar
    - Indicamos el estimador que queremos usar (algoritmo o pipeline)
    - Indicamos un *evaluator* para verificar los resultados.
- Spark automáticamente, genera distintas muestras de train/test de nuestro dataFrame de datos y empieza a ejecutar el *tunning*, finalmente en base al *evaluator*, selecciona el modelo con los mejores resultados.

Exercise 1

Pokemon Classifier

Exercise 2

Load Models

Exercise 3

Tunning: Cross-Validation

KEEPCODING

Tech School

Madrid | Barcelona | Bogotá

Datos de contacto