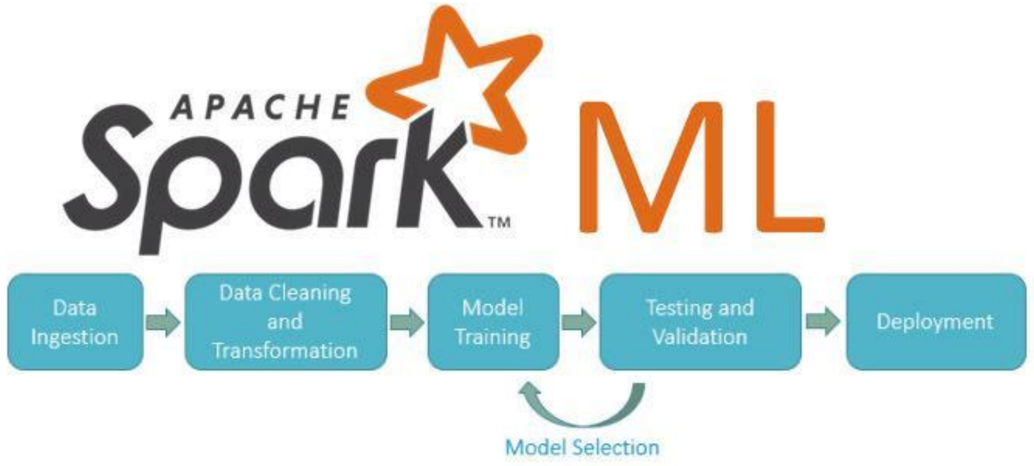
PySpark ML

Credit to Peerapon Vateekul

https://github.com/kaopanboonyuen/GISTDA2023

Outlines

- Overview
- What is Spark?
- Spark Machine Learning(ML) API
- Comparison between Spark ML and Spark Mllib
- Why Use Spark ML
- Programming Languages Supported by Spark ML
- Components of a SparkML Pipeline
- Spark ML Pipelines
- Algorithms Supported by Spark ML
- Performance
- Summary

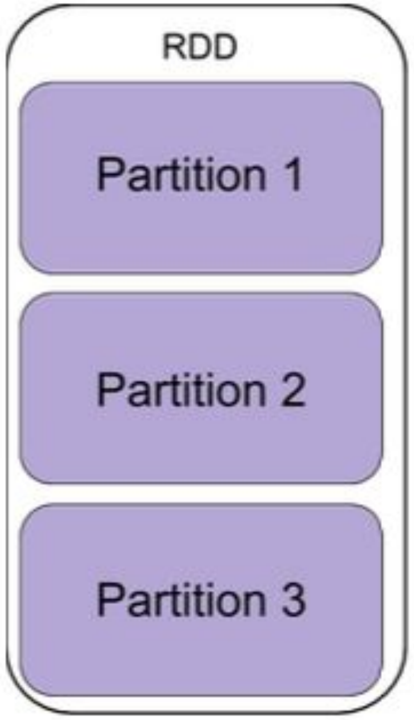


Overview

- The scalable machine learning library of Spark is MLlib.
- It contains general learning utilities and algorithms, which include regression, collaborative filtering, classification, clustering, dimensionality reduction, and underlying optimization primitives.
- There are two types of API available. The primary API is original spark. Machine learning library API **and** a higher-level API to construct Machine Learning workflows is Pipelines Spark Machine Learning.
- We will discuss Spark Machine Learning(ML) API in the next section

What is Spark?

- General purpose distributed system.
  - With a really nice API including Python .
- Apache project (one of the most active).
- Much faster than Hadoop Map/Reduce.
- Good when data is too big for a single machine .
- Built on top of two abstractions for distributed data: RDDs & Datasets



Spark Machine Learning(ML) API

- The APIs for machine learning algorithms are standardized by Spark ML.
- With this, it is easy to combine various algorithms in a single workflow or pipeline.
- The key concepts related to Spark ML API are listed below.
- The first concept is of an ML dataset.
- Spark machine learning utilizes the Spark SQL DataFrame as a dataset.
- It can contain various types of data types; for example, a dataset can contain different columns that store feature vectors, predictions, true labels, and text.

Spark Machine Learning(ML) API (cont.)

- An estimator is another algorithm that can produce a transformer fitting on a DataFrame. For instance, a learning algorithm can train on a dataset and produce a model.
- A pipeline specifies an ML workflow by chaining various transformers and estimators together.
- Param is the common API to specify parameters for all transformers and estimators.

Comparison between Spark ML and Spark MLlib

- A few key features of Spark ML and Spark MLlib are represented in a tabular format below this table.

Spark ML	Spark MLlib
Built on top of Spark DataFrame	Built on top of Spark RDD
Use pipeline to make combining multiple algorithms easy	Combining multiple algorithms is difficult
Can leverage catalyst engine optimization for SQL	Currently, supports more algorithms than Spark ML

Why Use Spark ML

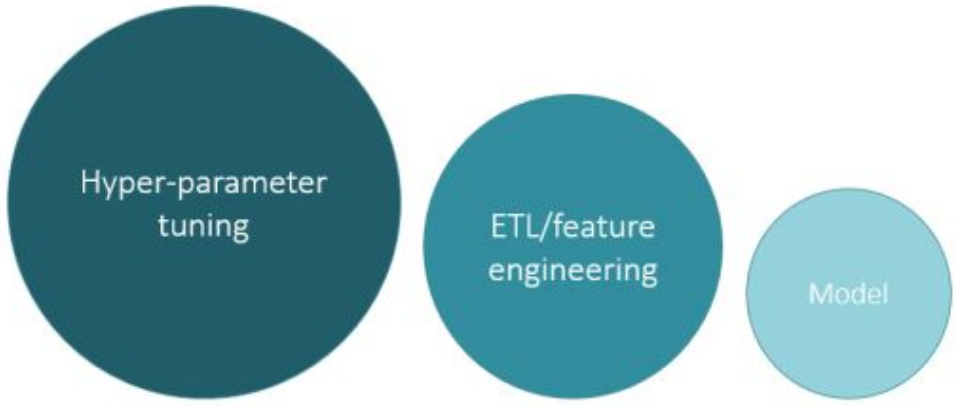
- Apart from the reasons mentioned in the previous section, there are several other reasons to use Spark ML over Spark MLlib.
- The sections below cover this in some detail.
  - 1. Solution to Machine Learning Scaling Challenges.
  - 2. Easy Data Transformation Enabled by DataFrame.
  - 3. Higher Speed of Execution Enabled by DataFrame.
  - 4. Ease of Creating Pipelines.
  - 5. The Future of Spark Machine Learning.

Why Use Spark ML

1. Solution to Machine Learning Scaling Challenges

- There are certain machine learning scaling challenges Spark ML tends to solve, that cannot be effectively solved by other libraries including Spark MLlib.
- These challenges are showcased in the figure below.
- The size of circle is directly proportional to the size of the challenge.
- It is easy to see that we have classified Hyper-parameter tuning as the most challenging task.

Challenges with scaling machine learning capabilities.

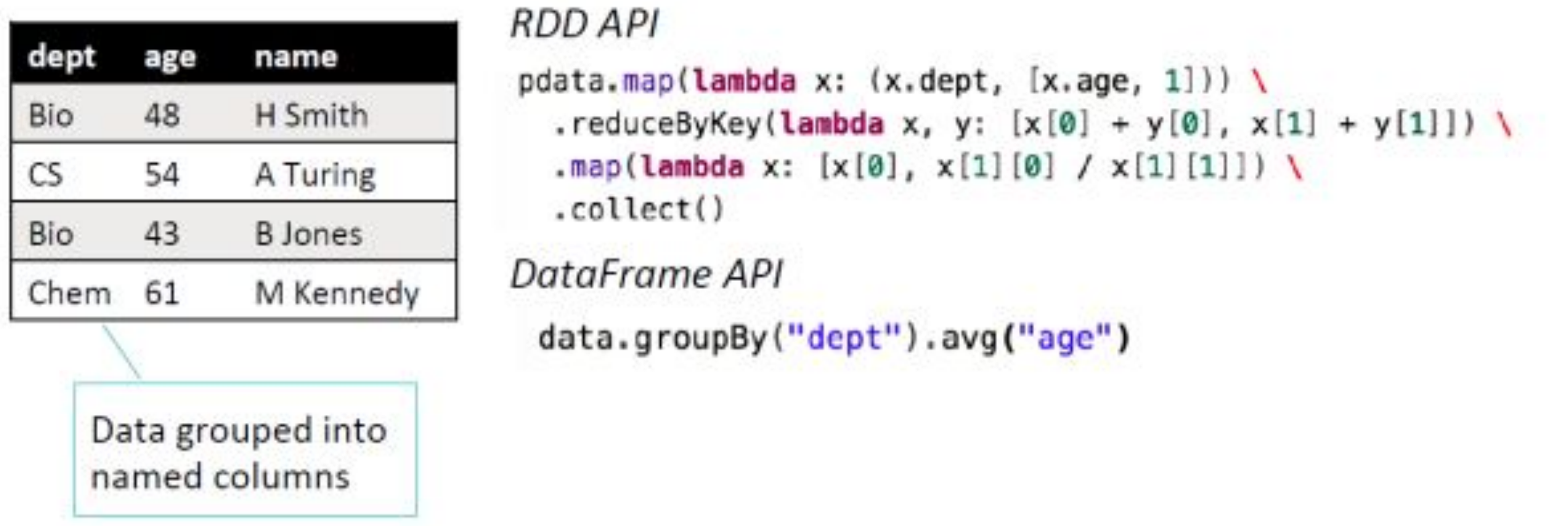


Why Use Spark ML

2. Easy Data Transformation Enabled by DataFrame

- Spark ML is easy to use for common data transformation tasks such as projection, filtering, aggregation and joining.
- We explain this with the help of an example below

Dataset for analysis and python code to calculate average age by department.



dept	age	name
Bio	48	H Smith
CS	54	A Turing
Bio	43	B Jones
Chem	61	M Kennedy

Data grouped into named columns

RDD API

```
pdata.map(lambda x: (x.dept, [x.age, 1])) \
  .reduceByKey(lambda x, y: [x[0] + y[0], x[1] + y[1]]) \
  .map(lambda x: [x[0], x[1][0] / x[1][1]]) \
  .collect()
```

DataFrame API

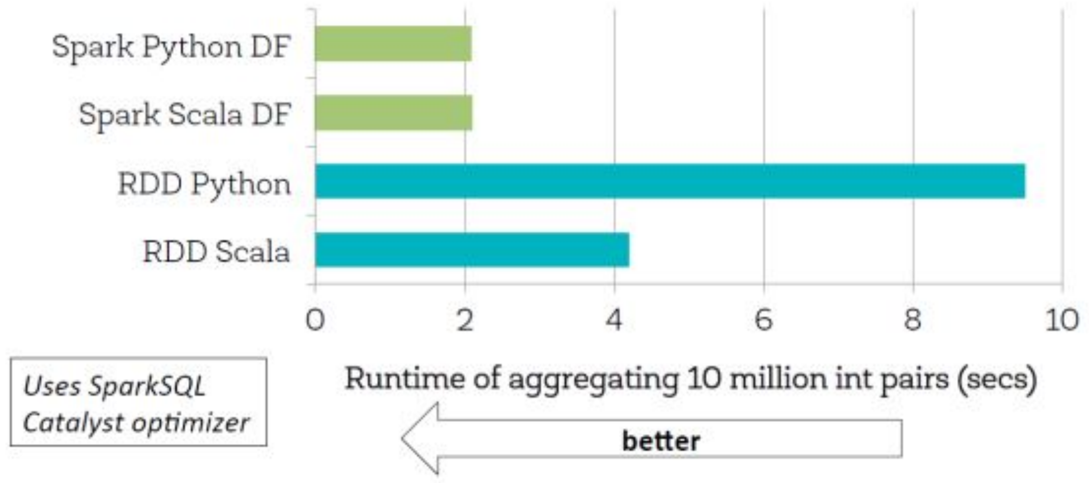
```
data.groupBy("dept").avg("age")
```

Why Use Spark ML

3. Higher Speed of Execution Enabled by DataFrame

- As introduced in section 2.1. DataFrames leverage the Catalyst Engine Optimization of SQLContext to perform operations in DataFrame objects.
- You will see that Spark DF (DataFrame) API for Python performs the aggregation five times faster than the Python RDD API, whereas the Scala DF API performs the same operation almost twice as fast as the Scala API for RDDs.

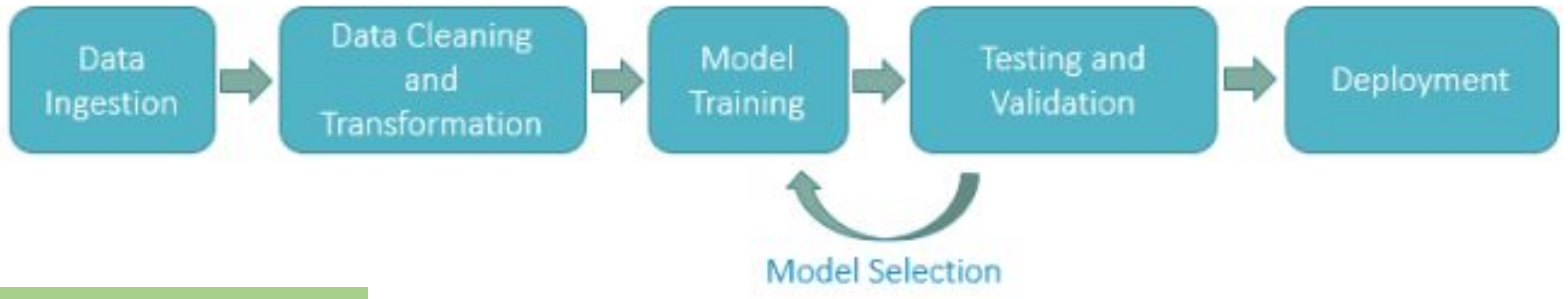
Run time comparison between different APIs for performing the same task



Why Use Spark ML

4. Ease of Creating Pipelines

- A practical ML pipeline often involves a sequence of data pre-processing, feature extraction, model fitting, and validation stages.
- Though there are many libraries we can use for each stage, connecting the dots is not as easy as it may look, especially with large-scale datasets.

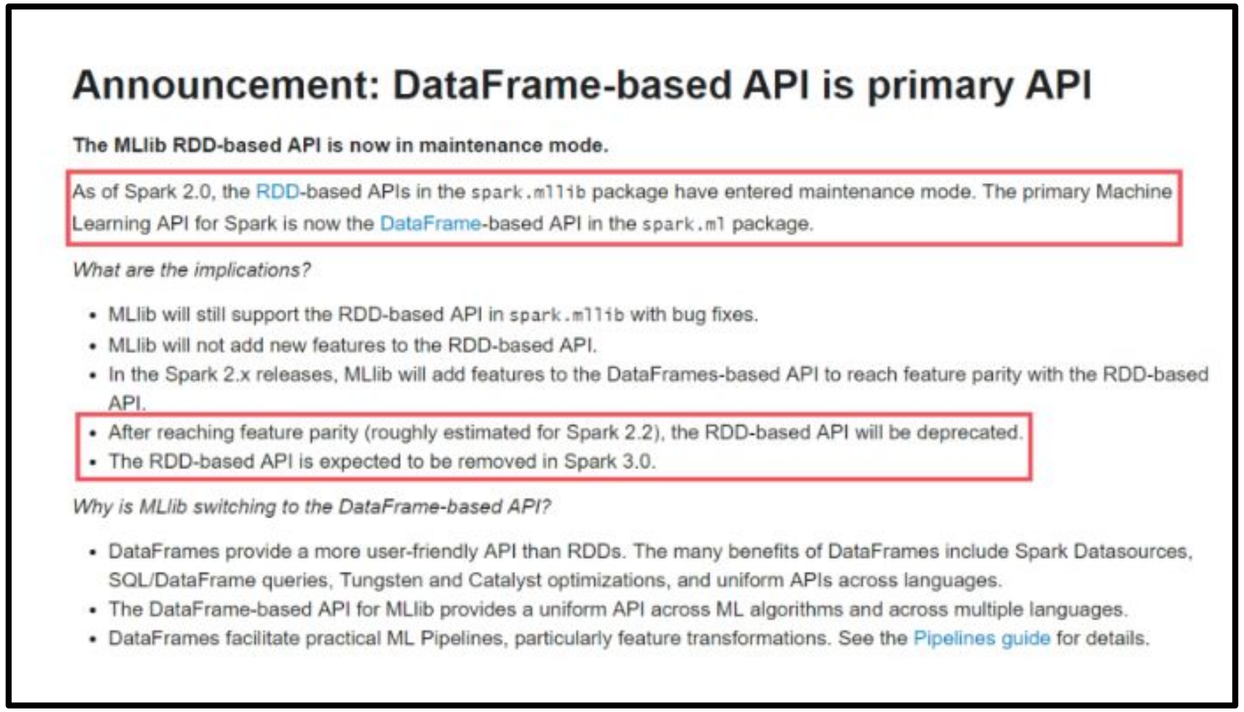


A typical Spark ML pipeline.

Why Use Spark ML

5. The Future of Spark Machine Learning

- Apart from the obvious technical benefits that Spark ML has, there is another important reason to shift from Spark MLlib to Spark ML.
- Spark MLlib has been of great importance in scalable enabling machine learning on Spark.
- However, time has come when MLlib has to be put on the back burner.
- As of the current version of Spark, MLlib is now in maintenance mode.
- A snapshot of the announcement posted on the official Spark website is shown in



Announcement: DataFrame-based API is primary API

The MLlib RDD-based API is now in maintenance mode.

As of Spark 2.0, the RDD-based APIs in the `spark.mllib` package have entered maintenance mode. The primary Machine Learning API for Spark is now the DataFrame-based API in the `spark.ml` package.

What are the implications?

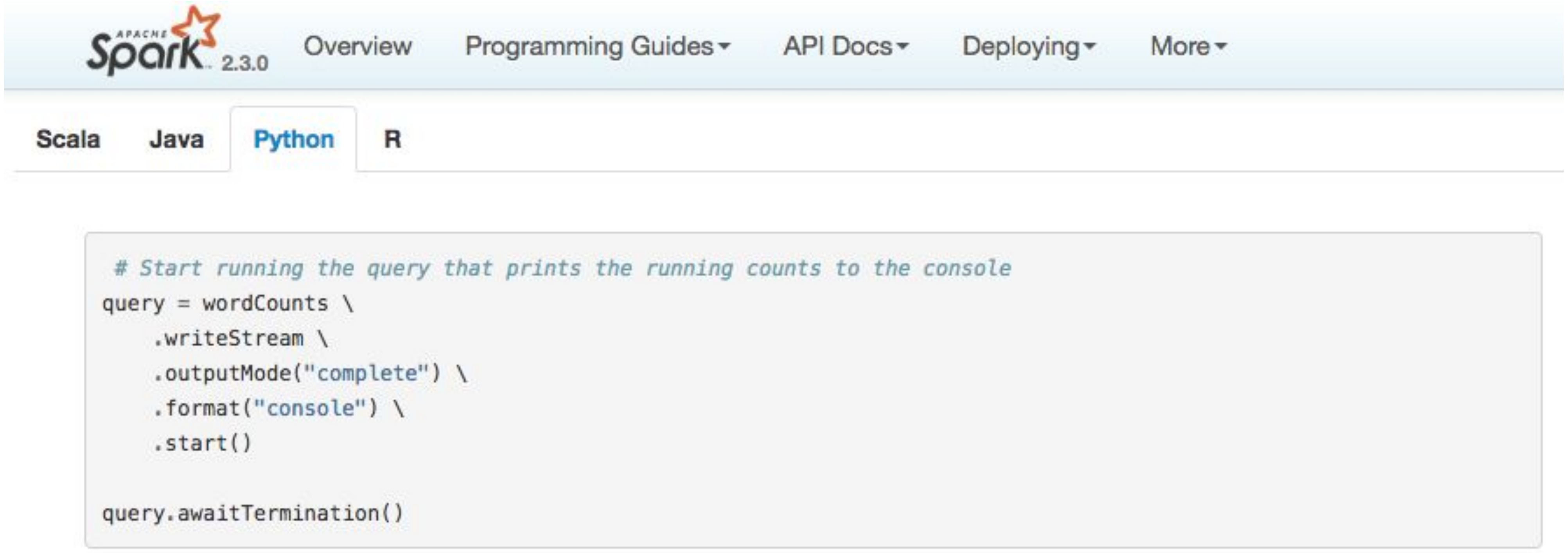
- MLlib will still support the RDD-based API in `spark.mllib` with bug fixes.
- MLlib will not add new features to the RDD-based API.
- In the Spark 2.x releases, MLlib will add features to the DataFrames-based API to reach feature parity with the RDD-based API.
- After reaching feature parity (roughly estimated for Spark 2.2), the RDD-based API will be deprecated.
- The RDD-based API is expected to be removed in Spark 3.0.

Why is MLlib switching to the DataFrame-based API?

- DataFrames provide a more user-friendly API than RDDs. The many benefits of DataFrames include Spark Datasources, SQL/DataFrame queries, Tungsten and Catalyst optimizations, and uniform APIs across languages.
- The DataFrame-based API for MLlib provides a uniform API across ML algorithms and across multiple languages.
- DataFrames facilitate practical ML Pipelines, particularly feature transformations. See the Pipelines guide for details.

Programming Languages Supported by Spark ML

- As of now three languages are supported by the API – Python, Scala, R and JAVA.



```
# Start running the query that prints the running counts to the console
query = wordCounts \
    .writeStream \
    .outputMode("complete") \
    .format("console") \
    .start()

query.awaitTermination()
```

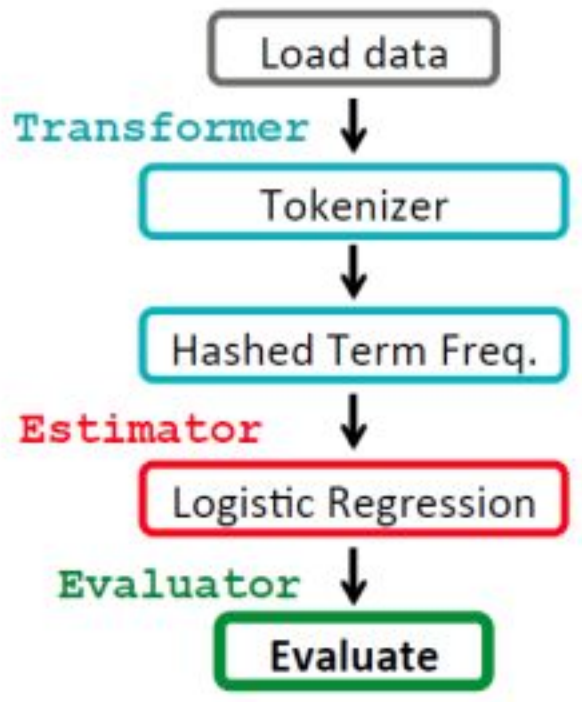
Components of a SparkML Pipeline

- (1) **DataFrame**:
  - Spark ML uses DataFrame rather than regular RDD as they hold a variety of data types (e.g. feature vectors, true labels, and predictions).
- (2) **Transformer**:
  - a transformer converts a DataFrame into another DataFrame usually by appending columns. (since Spark DataFrame is immutable, it actually creates a new DataFrame).
  - The implement method for a transformer is "transform()".
- (3) **Estimator**:
  - An Estimator is an algorithm which can be fit on a DataFrame to produce a Transformer.
  - Implements method fit() taking a DataFrame and a model (also a transformer) as input.

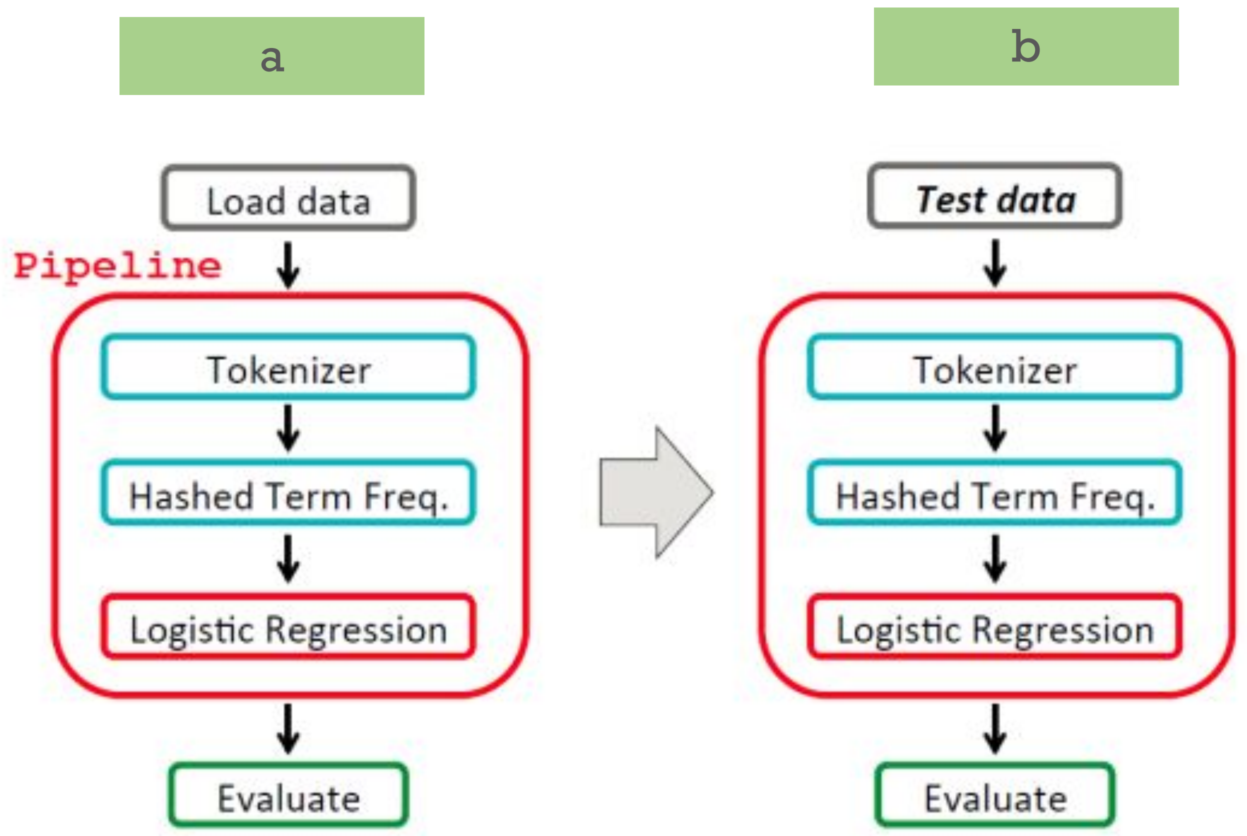
Components of a SparkML Pipeline (cont.)

- (4) **Pipeline**:
  - Chains multiple Transformers and Estimators each as a stage to specify an ML workflow.
  - These stages are run in order, and the input DataFrame is transformed as it passes through each stage.
- (5) **Parameter**:
  - All Transformers and Estimators now share a common API for specifying parameters.
- (6) **Evaluator:**
  - Evaluate model performance.
  - The Evaluator can be a RegressionEvaluator for regression problems, a BinaryClassificationEvaluator for binary data, or a MulticlassClassificationEvaluator for multiclass problems.

Components of a SparkML Pipeline (cont.)



A generic flow of machine learning components.

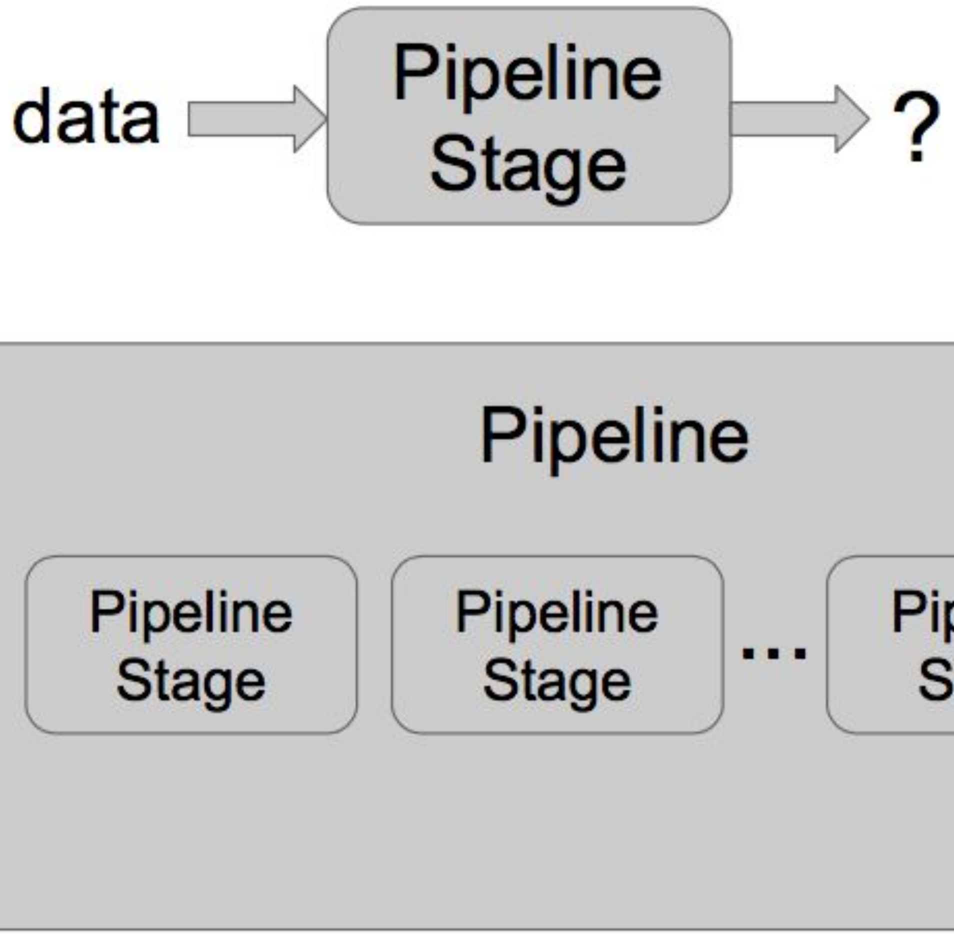


a) Pipeline of machine learning components.
b) Reusing the pipeline on Test phase.

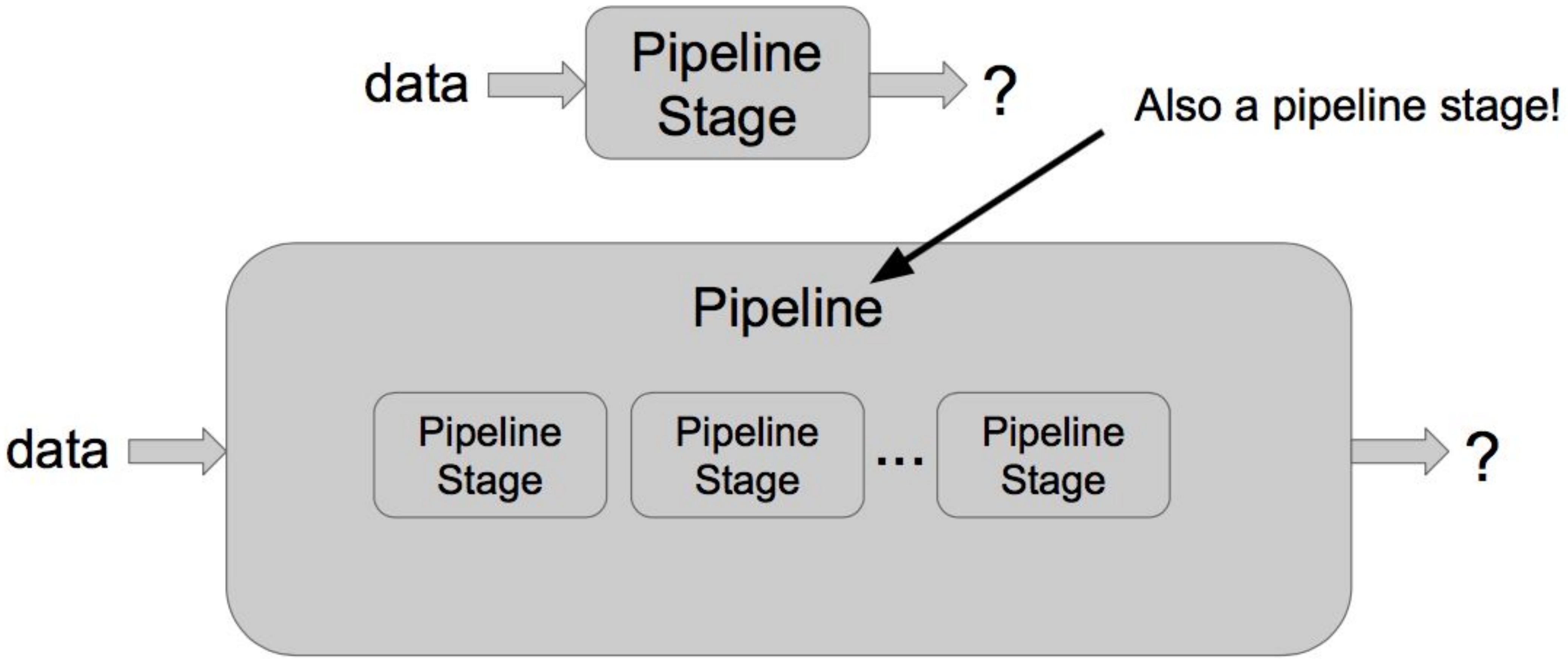
(Optional) Hyper Parameter Tuning

- Hyper-parameter tuning uses CrossValidator or TrainValidationSplit to select the Model produced by the best-performing set of parameters. The tool requires:
- **Estimator**:
  - algorithm or Pipeline to tune.
- **Set of ParamMaps**:
  - parameters to choose from, sometimes called a "parameter grid" to search over.
- **Evaluator**:
  - metric to measure how well a fitted Model does on held-out test data.

Spark ML Pipelines

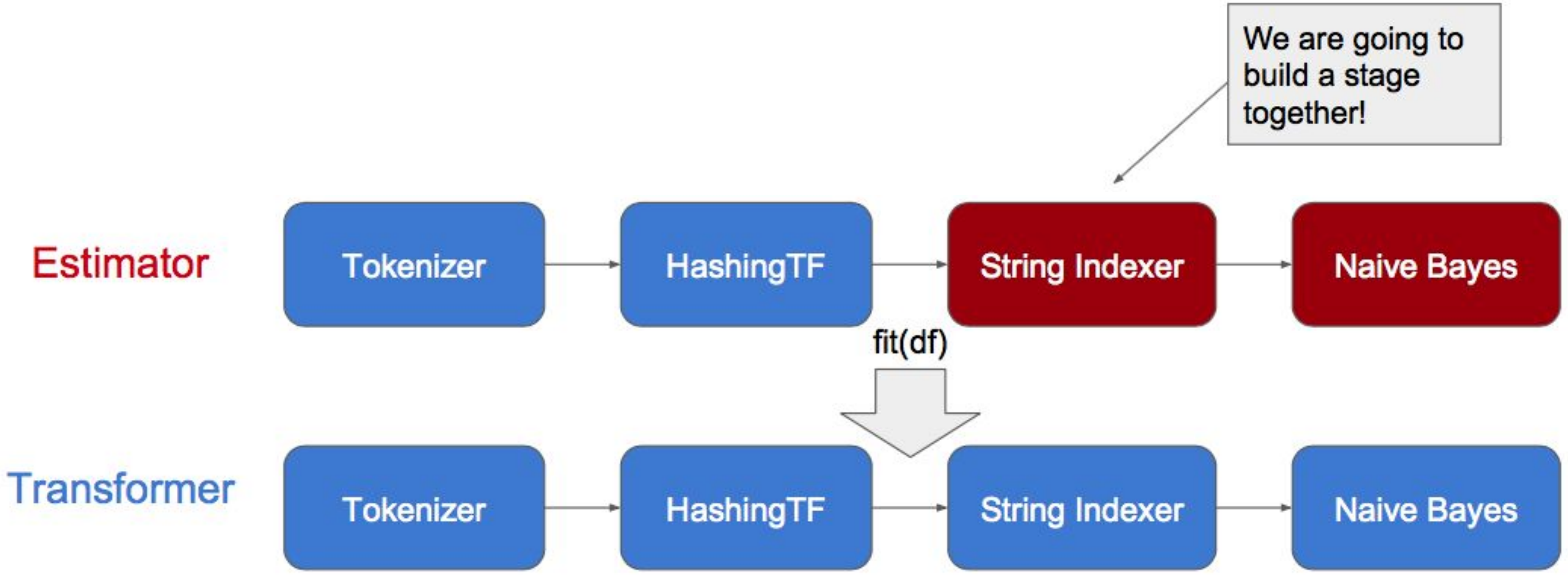


Spark ML Pipelines (cont.)

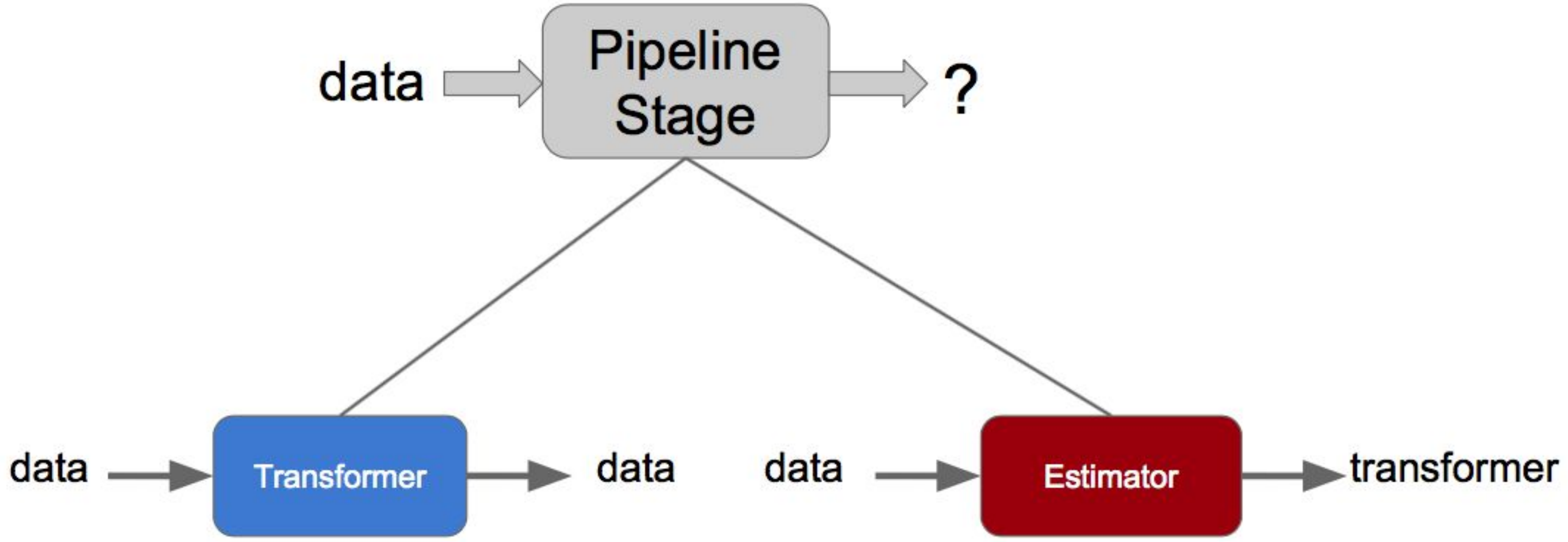


Spark ML Pipelines (cont.)

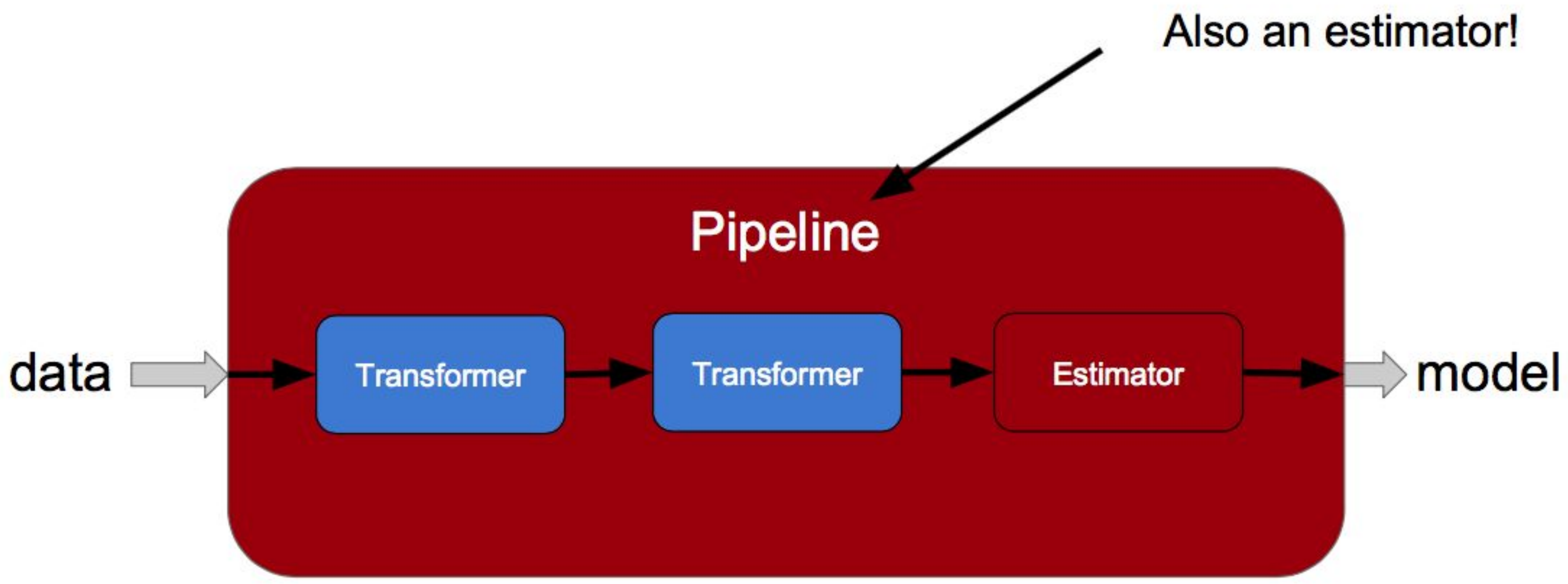
- Consist of different stages (estimators or transformers)
- Themselves are an estimator



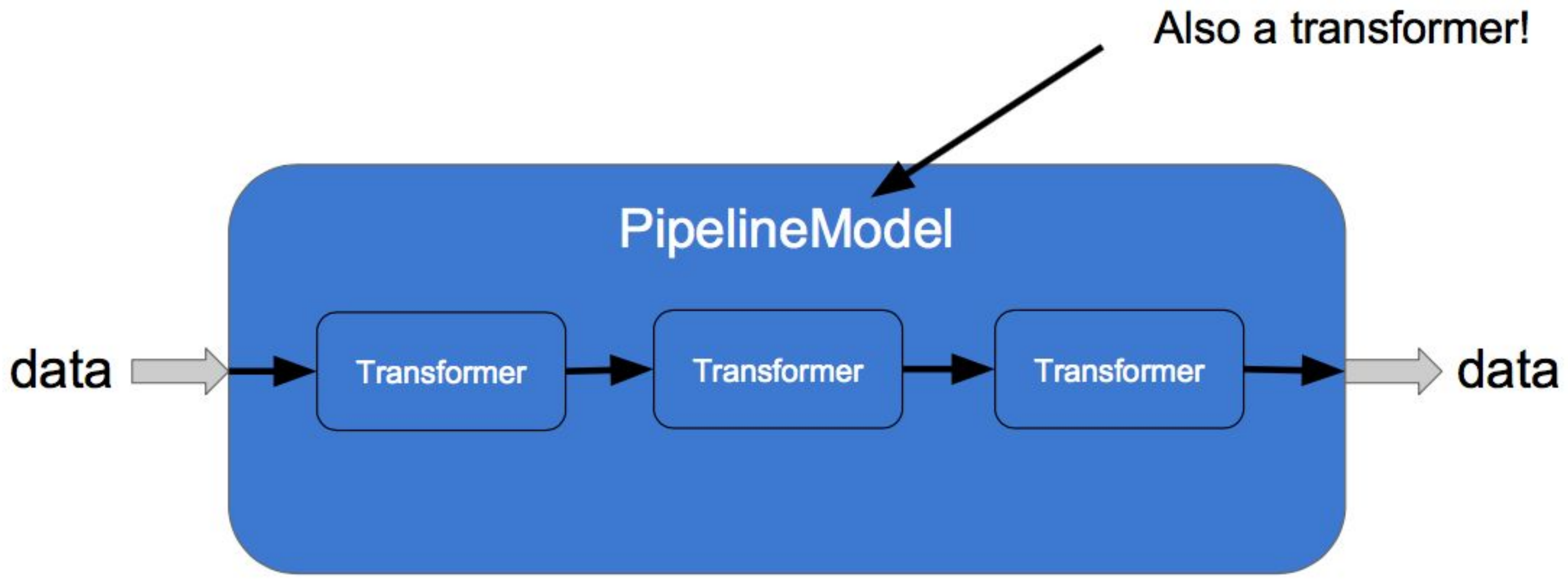
Two main types of pipeline stages



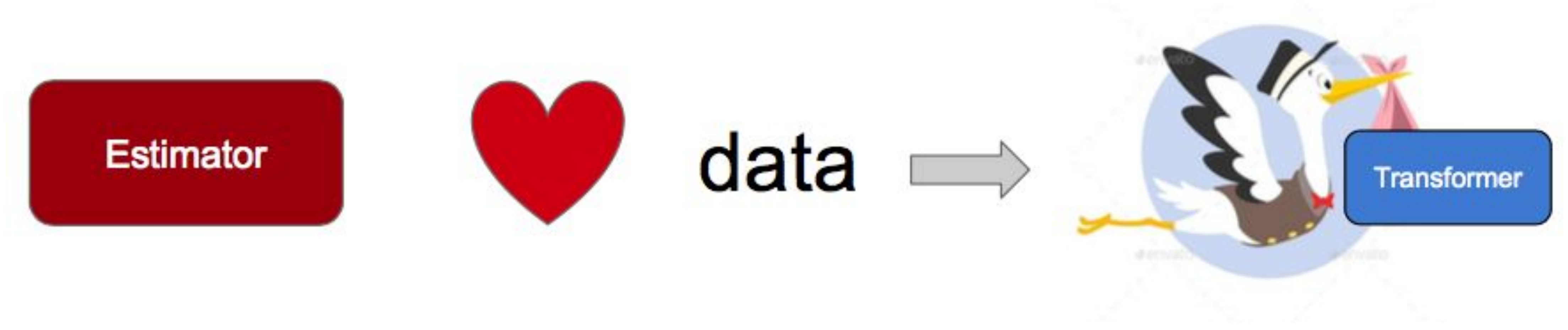
Pipelines are estimators



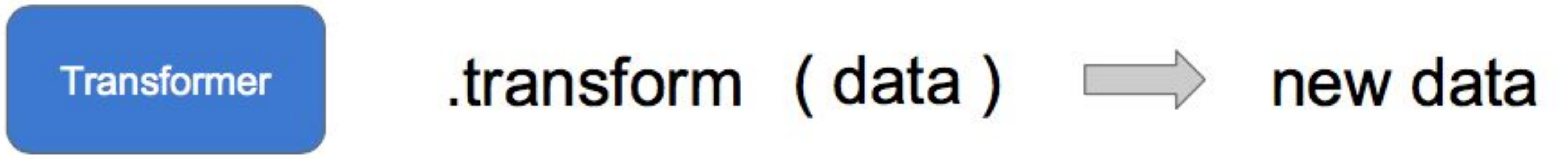
Pipeline Models are transformers



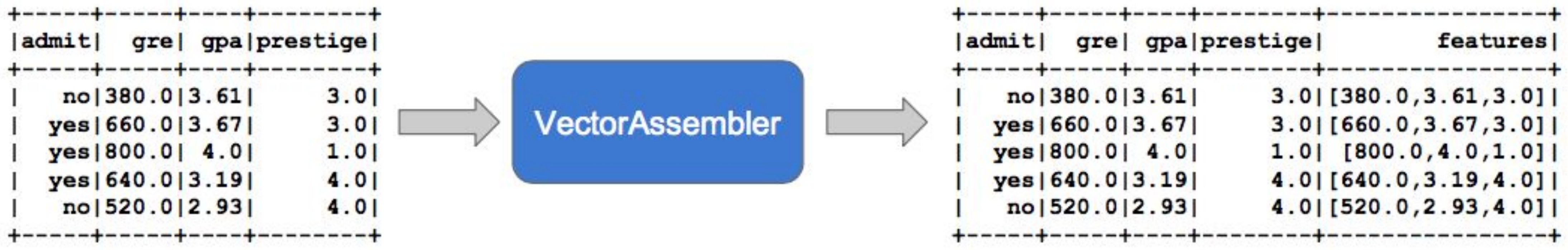
How are transformers made?



How is new data made?



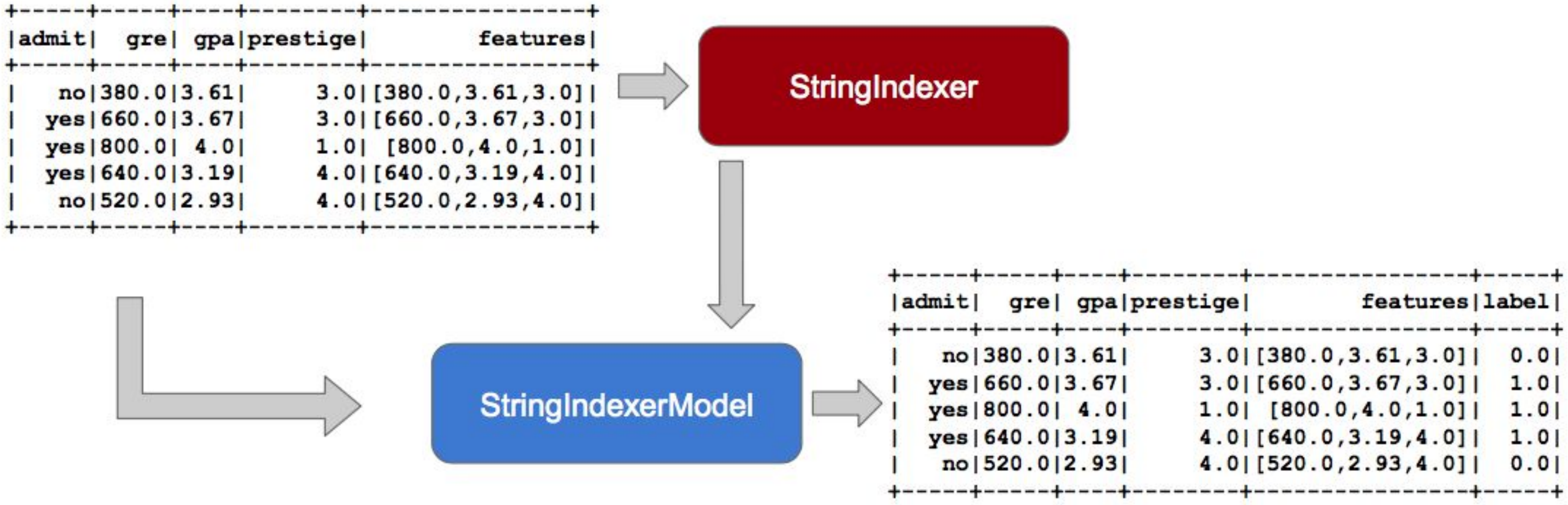
Feature transformations



```
+-----+-----+----+--------+
|admit|  gre| gpa|prestige|
+-----+-----+----+--------+
|   no|380.0|3.61|     3.0|
|  yes|660.0|3.67|     3.0|
|  yes|800.0| 4.0|     1.0|
|  yes|640.0|3.19|     4.0|
|   no|520.0|2.93|     4.0|
+-----+-----+----+--------+
```

```
+-----+-----+----+--------+----------------+
|admit|  gre| gpa|prestige|        features|
+-----+-----+----+--------+----------------+
|   no|380.0|3.61|     3.0|[380.0,3.61,3.0]|
|  yes|660.0|3.67|     3.0|[660.0,3.67,3.0]|
|  yes|800.0| 4.0|     1.0| [800.0,4.0,1.0]|
|  yes|640.0|3.19|     4.0|[640.0,3.19,4.0]|
|   no|520.0|2.93|     4.0|[520.0,2.93,4.0]|
+-----+-----+----+--------+----------------+
```

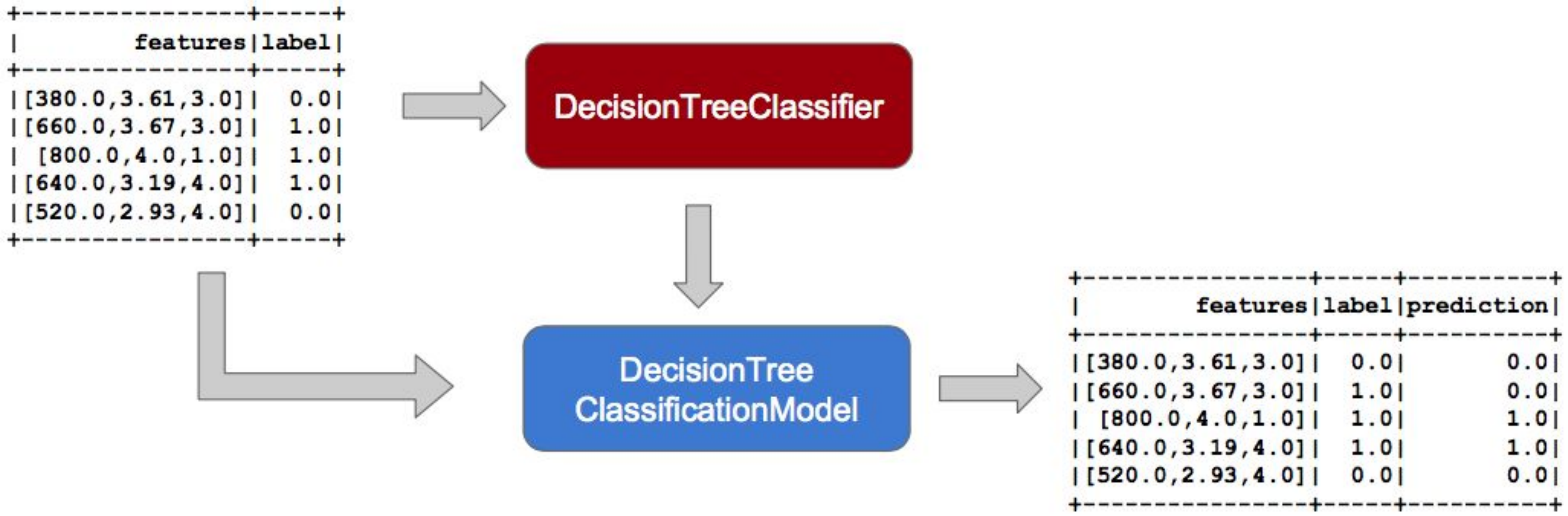
Train a classifier on the transformed data



```
+-----+-----+----+--------+----------------+
|admit|  gre| gpa|prestige|        features|
+-----+-----+----+--------+----------------+
|   no|380.0|3.61|     3.0|[380.0,3.61,3.0]|
|  yes|660.0|3.67|     3.0|[660.0,3.67,3.0]|
|  yes|800.0| 4.0|     1.0| [800.0,4.0,1.0]|
|  yes|640.0|3.19|     4.0|[640.0,3.19,4.0]|
|   no|520.0|2.93|     4.0|[520.0,2.93,4.0]|
+-----+-----+----+--------+----------------+
```

```
+-----+-----+----+--------+----------------+-----+
|admit|  gre| gpa|prestige|        features|label|
+-----+-----+----+--------+----------------+-----+
|   no|380.0|3.61|     3.0|[380.0,3.61,3.0]|  0.0|
|  yes|660.0|3.67|     3.0|[660.0,3.67,3.0]|  1.0|
|  yes|800.0| 4.0|     1.0| [800.0,4.0,1.0]|  1.0|
|  yes|640.0|3.19|     4.0|[640.0,3.19,4.0]|  1.0|
|   no|520.0|2.93|     4.0|[520.0,2.93,4.0]|  0.0|
+-----+-----+----+--------+----------------+-----+
```

Train a classifier on the transformed data

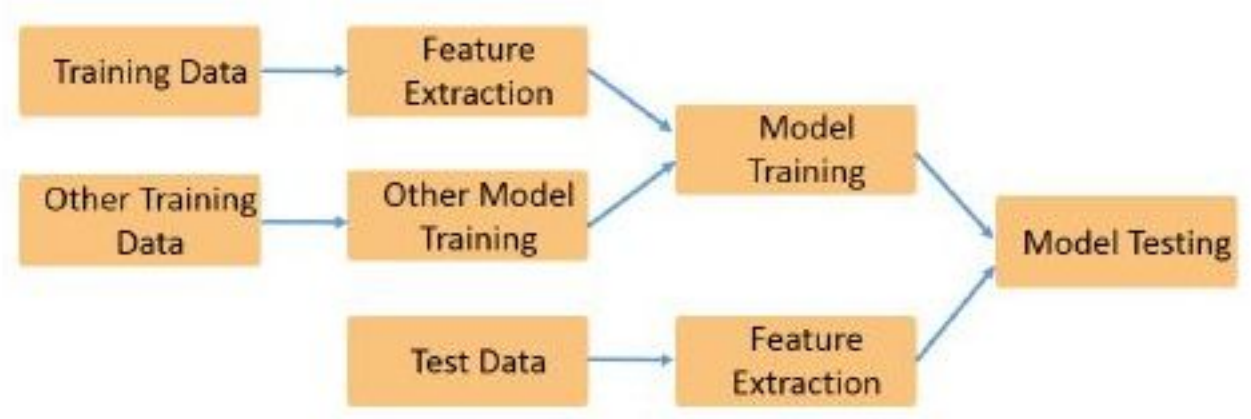


```
+----------------+-----+
|        features|label|
+----------------+-----+
|[380.0,3.61,3.0]|  0.0|
|[660.0,3.67,3.0]|  1.0|
| [800.0,4.0,1.0]|  1.0|
|[640.0,3.19,4.0]|  1.0|
|[520.0,2.93,4.0]|  0.0|
+----------------+-----+
```

```
+----------------+-----+----------+
|        features|label|prediction|
+----------------+-----+----------+
|[380.0,3.61,3.0]|  0.0|       0.0|
|[660.0,3.67,3.0]|  1.0|       0.0|
| [800.0,4.0,1.0]|  1.0|       1.0|
|[640.0,3.19,4.0]|  1.0|       1.0|
|[520.0,2.93,4.0]|  0.0|       0.0|
+----------------+-----+----------+
```

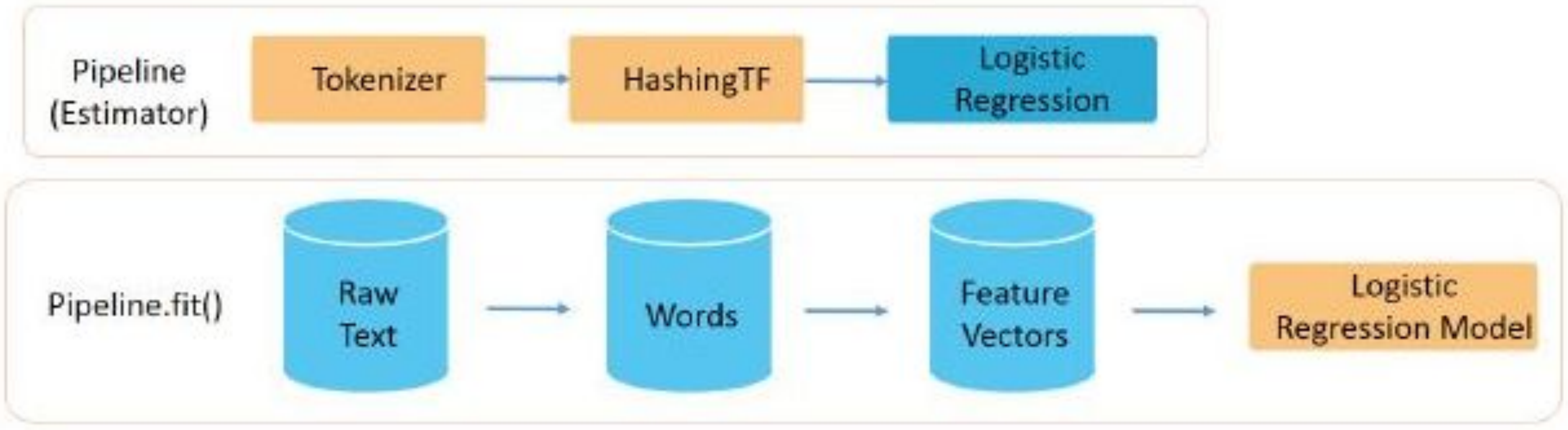
Pipeline

- Running an algorithms sequence for processing and learning from data is common in machine learning.
- For instance, the process workflow of a simple text document includes the stages listed below. First, the text of every document is split into words.
- Then, these words are converted into a numerical feature vector. And, at the final stage, the feature vectors and labels are used to learn a prediction model.

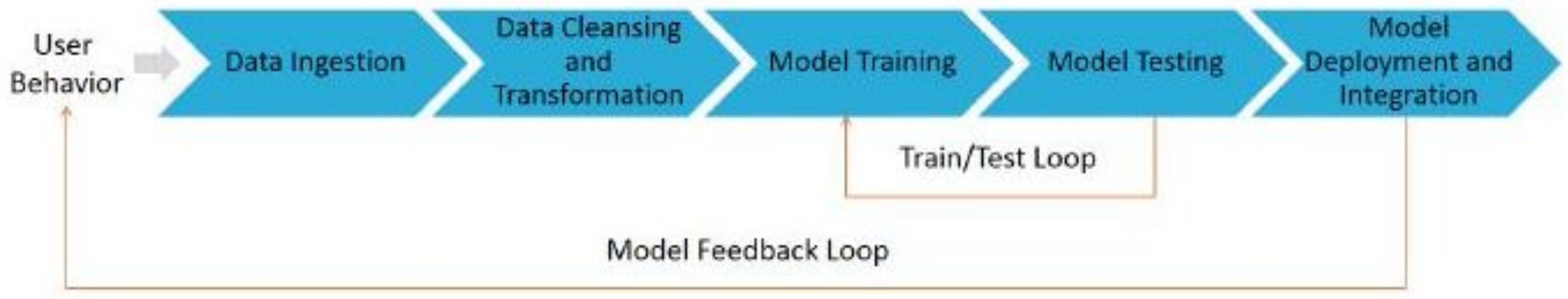


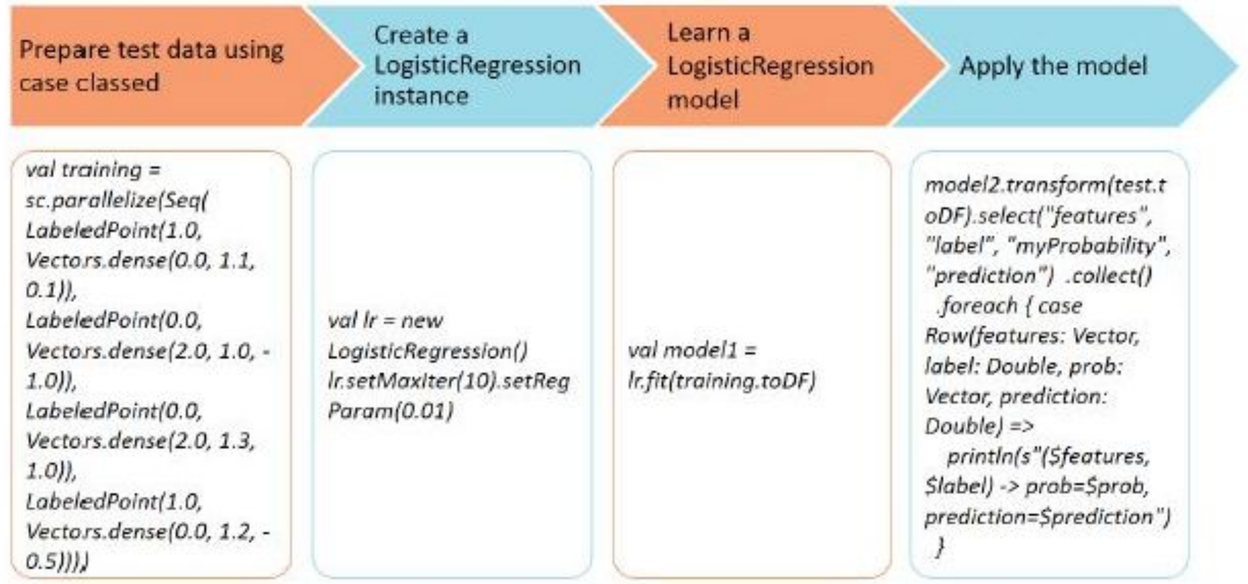
Working of a Pipeline

- This pipeline has three stages, two of which, Tokenizer and HashingTF, are transformers, while the third, LogisticRegression, is an estimator.
- The row below it shows the flow of data through the pipeline, where the cylinders represent DataFrames.



General Machine Learning Pipeline-Example




Prepare test data using case classed
Create a LogisticRegression instance
Learn a LogisticRegression model
Apply the model

```
val training =
sc.parallelize(Seq(
LabeledPoint(1.0,
Vectors.dense(0.0, 1.1,
0.1)),
LabeledPoint(0.0,
Vectors.dense(2.0, 1.0, -
1.0)),
LabeledPoint(0.0,
Vectors.dense(2.0, 1.3,
1.0)),
LabeledPoint(1.0,
Vectors.dense(0.0, 1.2, -
0.5))))
```

```
val lr = new
LogisticRegression()
lr.setMaxIter(10).setReg
Param(0.01)
```

```
val model1 =
lr.fit(training.toDF)
```

```
model2.transform(test.t
oDF).select("features",
"label", "myProbability",
"prediction")  .collect()
 .foreach { case
Row(features: Vector,
label: Double, prob:
Vector, prediction:
Double) =>
   println(s"($features,
$label) -> prob=$prob,
prediction=$prediction")
 }
```

Algorithms Supported by Spark ML

- Machine learning with SparkML is scalable and easy.
- Classification
  - Logistic regression, Decision Tree Classifier, Random Forest Classifier, Gradient-Boosted Tree Classifier, Multilayer Perceptron Classifier, One-vs-Rest Classifier (a.k.a. One-vs-All), Naive Bayes.
- Numeric Predictions
  - Linear Regression, Generalized Linear Regression, Available Families, Decision Tree Regression, Random Forest Regression, Gradient-Boosted Tree Regression, Survival Regression, Isotonic Regression.

Algorithms Supported by Spark ML (Cont.)

- Machine learning with SparkML is scalable and easy.
- Clustering
  - K-Means, Latent Dirichlet Allocation (LDA), Bisecting K-Means, Gaussian Mixture Model (GMM).
- Collaborative filtering
  - Explicit Vs. Implicit Feedback, Scaling of the Regularization Parameter.
- Apart from machine learning algorithm Spark also offers tools for several different functions such as:
  - Featurization: feature extraction, transformation, dimensionality reduction, and selection
  - Pipelines: tools for constructing, evaluating, and tuning ML Pipelines
  - Persistence: saving and load algorithms, models, and Pipelines
  - Utilities: linear algebra, statistics, data handling, etc.

Algorithms Supported by Spark ML (Cont.)

- Few examples of (1) Feature Extractors, (2) Feature Transformers and (3) Feature Selectors are as follows:
- (1) Feature Extractors
  - TF-IDF
  - Word2Vec
  - CountVectorizer
- (2) Feature Selectors
  - VectorSlicer
  - Rformula
  - ChiSqSelector

Algorithms Supported by Spark ML (Cont.)

- (3) Feature Transformers
  - Tokenizer, StopWordsRemover, n-gram, Binarizer, PCA, PolynomialExpansion
  - Discrete Cosine Transform (DCT)
  - StringIndexer, IndexToString, OneHotEncoder, VectorIndexer, Normalizer
  - StandardScaler, MinMaxScaler, MaxAbsScaler, Bucketizer, ElementwiseProduct
  - SQLTransformer, VectorAssembler,
  - QuantileDiscretizer

Model Selection via Cross-Validation

- Model selection is an essential task in machine learning.
- It includes the use of data to figure out the best parameters or model for a function.
- It is also termed as tuning.
- Pipelines facilitate model selection, which does not tune each element in the pipeline separately and makes optimizing an entire pipeline in one go.

Data Types

- MLlib provides support to different data types listed on the screen.
- **A local vector** includes 0-based indices, integer-based indices, and double-typed values. These values are saved on a single machine.
  - There are two types of local vectors supported by MLlib, which are sparse and dense.
  - A sparse vector is backed up by two parallel arrays, which are values and indices.
  - On the other hand, a dense vector is backed by a double array that represents its entry values.
- **A labeled point** is also a local vector, but it is related to a label or a response.
  - These are utilized in supervised learning algorithms in MLlib.
  - To use labeled points in both classification and regression, a double is used to store a label.
  - In case of multiclass classification, labels should be class indices that start from zero.
  - However, in case of binary classification, these should be 0 or 1.
- **A local matrix** includes double-typed values, and integer-typed row and column indices.
  - Like local vectors, these values are also saved on a single machine.
  - In this case, only dense matrices are supported by MLlib.
  - The related entry values are saved in a single, double array in the column.

Feature Extraction and Basic Statistics

- Term Frequency-Inverse Document Frequency or TF-IDF is defined an easy way for generating feature vectors using text documents such as web pages.
- It is calculated as two statistics for every term in every document: TF that is defined as the number of times the term appears in the particular document.
- IDF that is defined as the measure how often a term appears in the entire document corpus.
- The product of TD per times IDF signifies the relevance of a term to a specific document.

Clustering

- Clustering is defined as an unsupervised learning problem in which the objective is to group the entities subsets by some idea of similarity.

- It is used as a hierarchical supervised learning pipeline component or for exploratory analysis.

- MLlib supports various models of clustering, which include
  - K-means
  - Gaussian mixture
  - Power iteration clustering or PIC
  - Latent Dirichlet allocation or LDA
  - Streaming k-means.

Collaborative Filtering

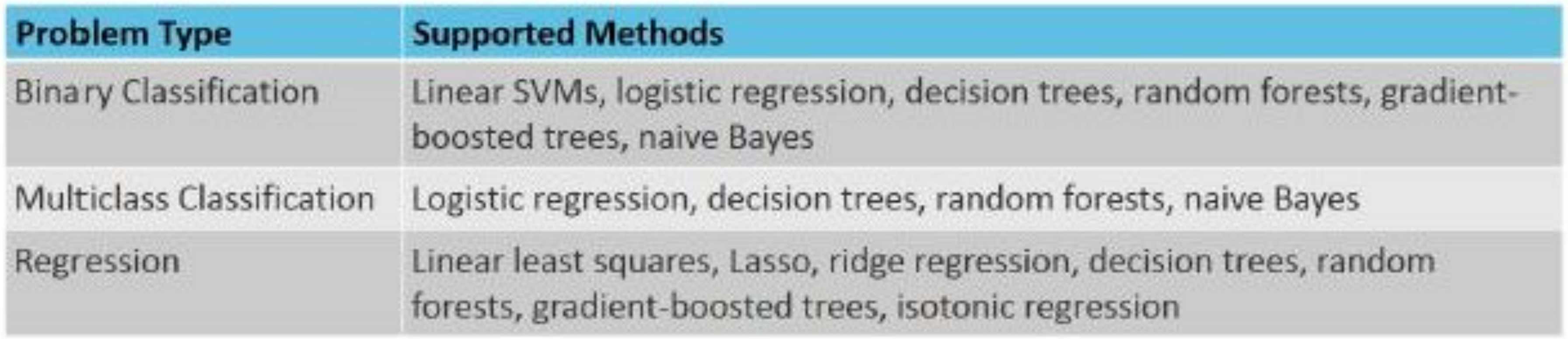
- Collaborative filtering is used for recommender systems.
- The objective of these techniques is to complete a user-term association matrix entries that are missing.
- At present, MLlib provides support to model-based collaborative filtering.
- In this, products and users have explained through a small latent factor set. These factors are used for predicting the missing entries.
- To learn these factors, MLlib utilizes the alternative least squares or ALS algorithm.
- The MLlib implementation includes the parameters listed next.

Collaborative Filtering (Cont.)

- numBlocks is defined as the number of blocks that are used for parallelizing computation and rank is defined as the number of latent factors in the model.
  - Iterations are defined as the number of iterations to run, while lambda defines the parameter of regularization in ALS.
- implicitPrefs defines what should be used out of the variant adapted for implicit feedback data or the explicit feedback ALS variant.
  - alpha is a parameter that applies to the implicit feedback ALS variant governing the baseline confidence in preference observation.

Classification

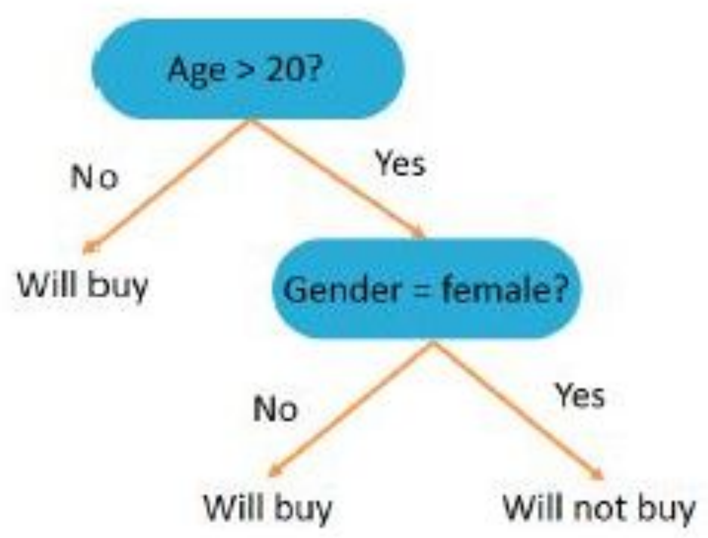
- MLlib provides support to different regression analysis, binary classification, and multiclass classification methods.
- The table shown below lists and explains the supported algorithms for every problem type.

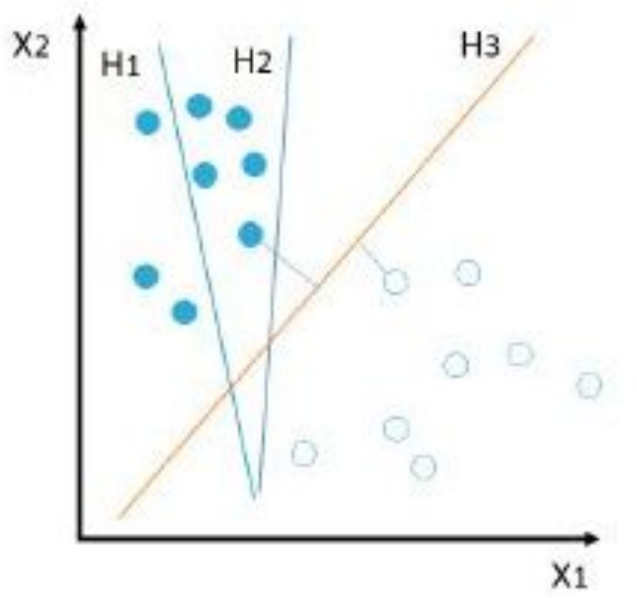


Problem Type	Supported Methods
Binary Classification	Linear SVMs, logistic regression, decision trees, random forests, gradient-boosted trees, naive Bayes
Multiclass Classification	Logistic regression, decision trees, random forests, naive Bayes
Regression	Linear least squares, Lasso, ridge regression, decision trees, random forests, gradient-boosted trees, isotonic regression

Classification (Cont.)

- Two examples of classification are shown below.

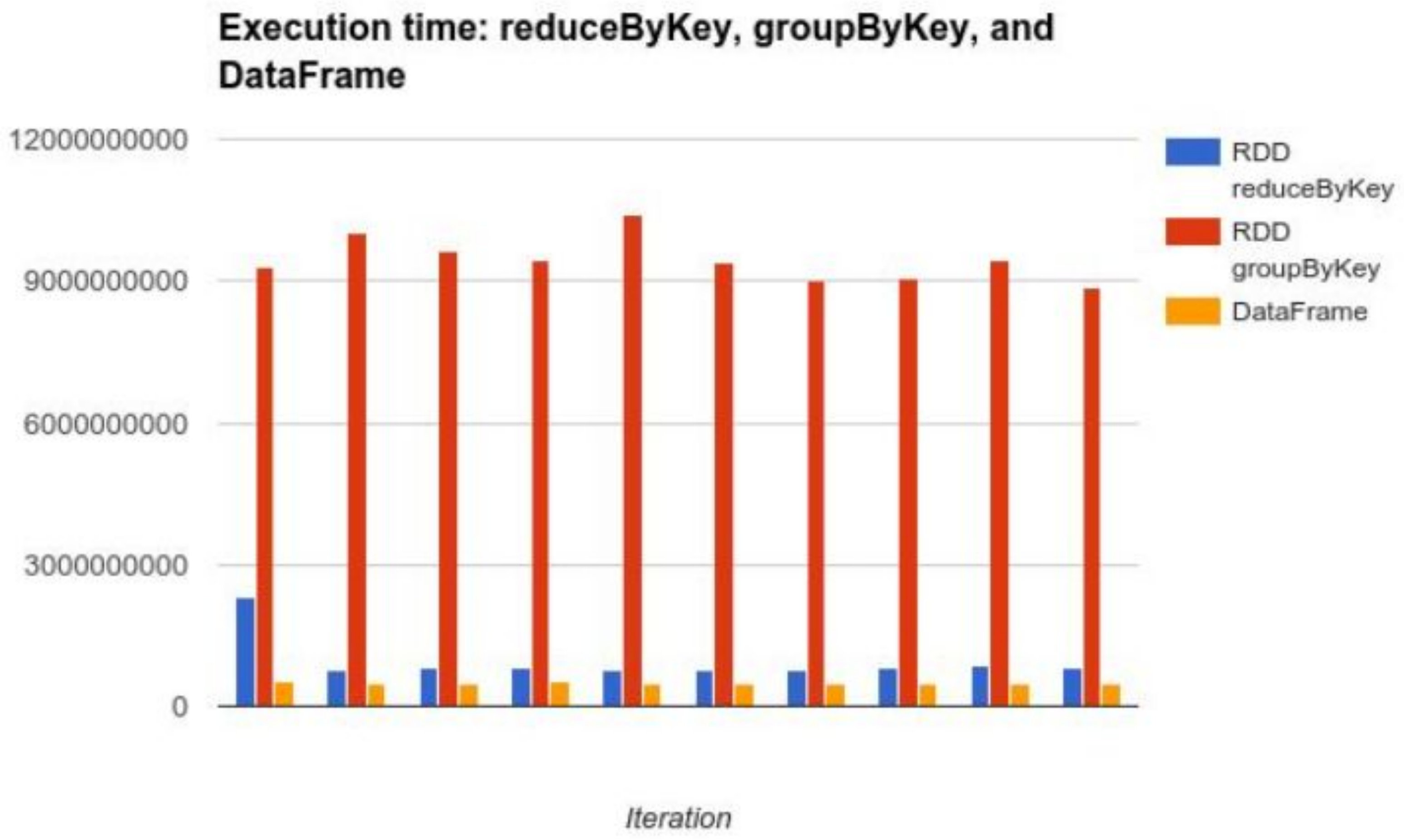




Regression

- In linear regression, multiple procedures have been developed for parameter inference and estimation.
- These procedures vary concerning the presence of a closed-form solution, robustness concerning heavy-tailed distributions, and computational simplicity of algorithms.
- They also differ to theoretical assumptions that are required for validating the desirable statistical properties like asymptotic efficiency and consistency.
- One of the statistical problem-solving methods is linear least squares.
- This allows increasing the accuracy of solution approximation to the complexity of a specific problem.

What is the performance like?



Summary

- Machine Learning is a subfield of Artificial Intelligence that has empowered various smart applications.
- Some terminologies used in ML are feature vector, feature space, samples, and labeled data.
- The scalable machine learning library of Spark is MLlib.
- Spark ML includes the Spark SQL DataFrame to support ML data types.
- A transformer consists of learned models and feature transformers.
- Workflows in Spark ML are represented through pipelines.

Summary (Cont.)

- A non-linear pipeline can also be created as long as the graph of data flow creates a DAG.
- Param is the uniform API to specify parameters for estimators and transformers.
- The Model selection includes the use of data to figure out the best parameters or model for a task.
- MLlib supports data types including local vector, labeled point, and local matrix.
- TF-IDF is defined an easy way for generating feature vectors using text documents such as web pages.
- Clustering is defined as an unsupervised learning problem in which the objective is to group the entities subsets by some idea of similarity.

Summary (Cont.)

- The Collaborative filter is used to complete a user-term association matrix entries that are missing.
- MLlib provides support to different regression analysis, binary classification, and multiclass classification methods.
- In linear regression, multiple procedures have been developed for parameter inference and estimation.

Reference

- https://yurongfan.wordpress.com/2017/01/10/introduction-of-a-big-data-machine-learning-tool-sparkml/
- https://www.simplilearn.com/spark-ml-programming-tutorial-video
- https://yurongfan.wordpress.com/2017/01/10/introduction-of-a-big-data-machine-learning-tool-sparkml/