



Spark PL

BigData. Фреймворк Apache Spark, урок 4



Spark ML vs MLlib



Spark MLlib

13

spark.mllib contains the legacy API built on top of RDDs.



Spark ML

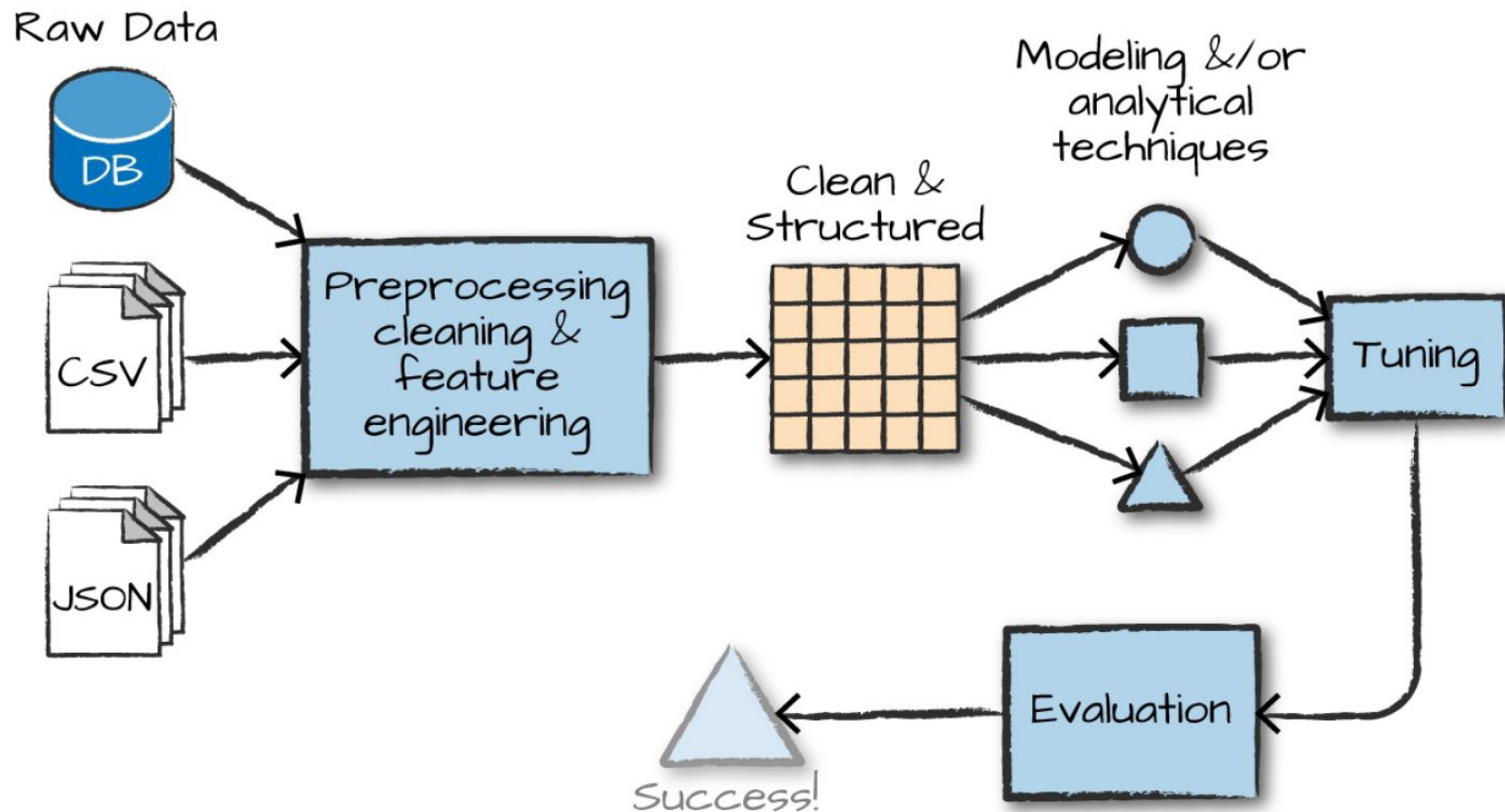


spark.ml provides higher-level API built on top of DataFrames for constructing ML pipelines.

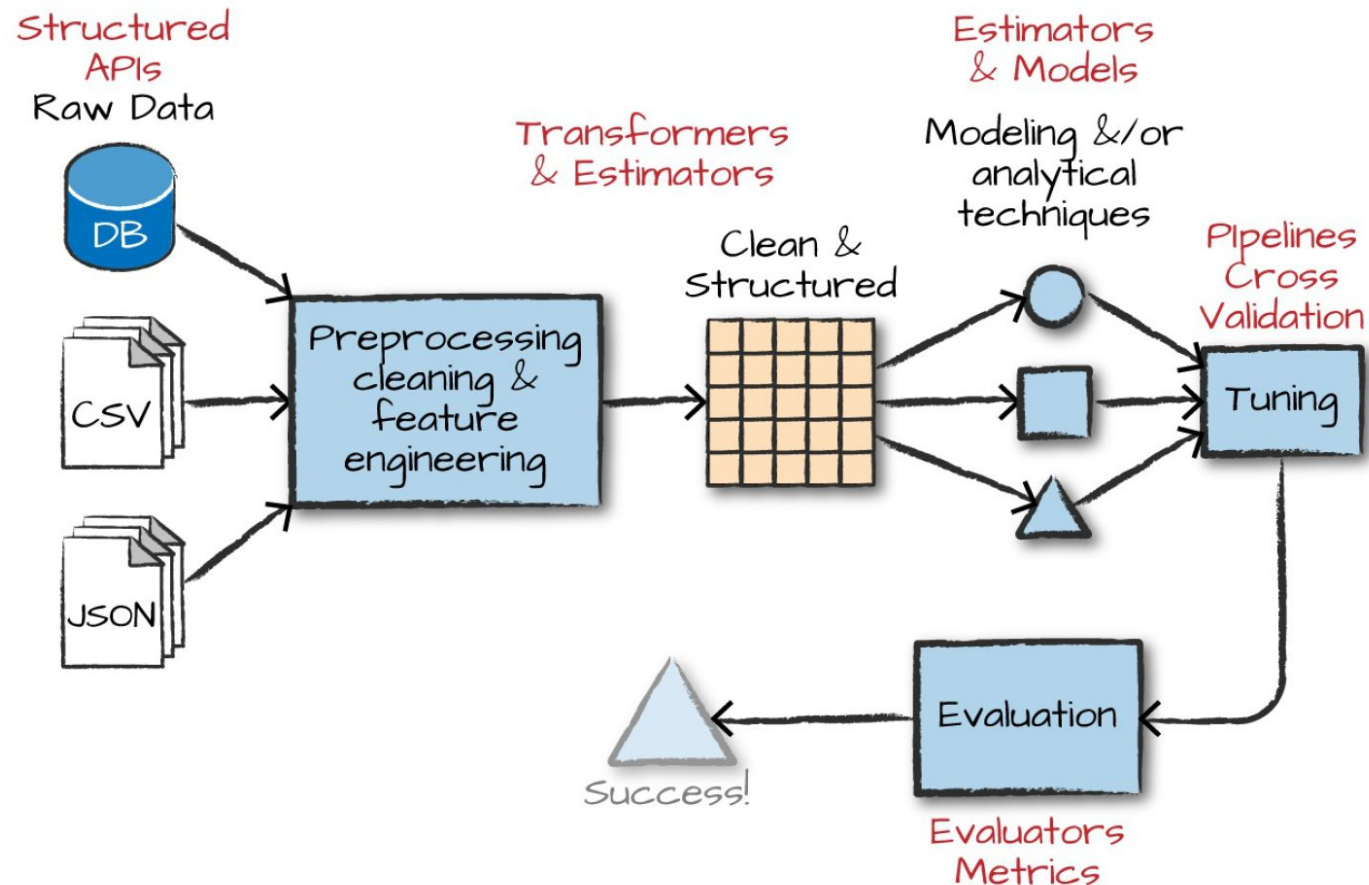
4 причины строить ML-pipeline, а не просто модели Machine Learning

- ⚡ **Чистый код** за счёт автоматизации процедур подготовки данных — выборка, очистка, генерация предикторов (фичей, от англ. feature) и пр.
- ⚡ **Сокращение ошибок** благодаря отработанной последовательности шагов (пропустить или неправильно выполнить какой-то этап не получится)
- ⚡ **Простота развёртывания в production** — обычно преобразовать ML-модель от прототипа к масштабируемому и надёжному решению для промышленной эксплуатации достаточно сложно, однако конвейеры помогут и здесь, облегчая тестирование и прочие MLOps-процедуры
- ⚡ **Дополнительная проверка ML-модели** — можно применить перекрёстную проверку (кросс-валидацию) и другие методы к этапам конвейера, пробуя различные параметры. Это ускоряет оптимизацию алгоритма и выбор наилучших конфигурационных настроек

The machine learning workflow



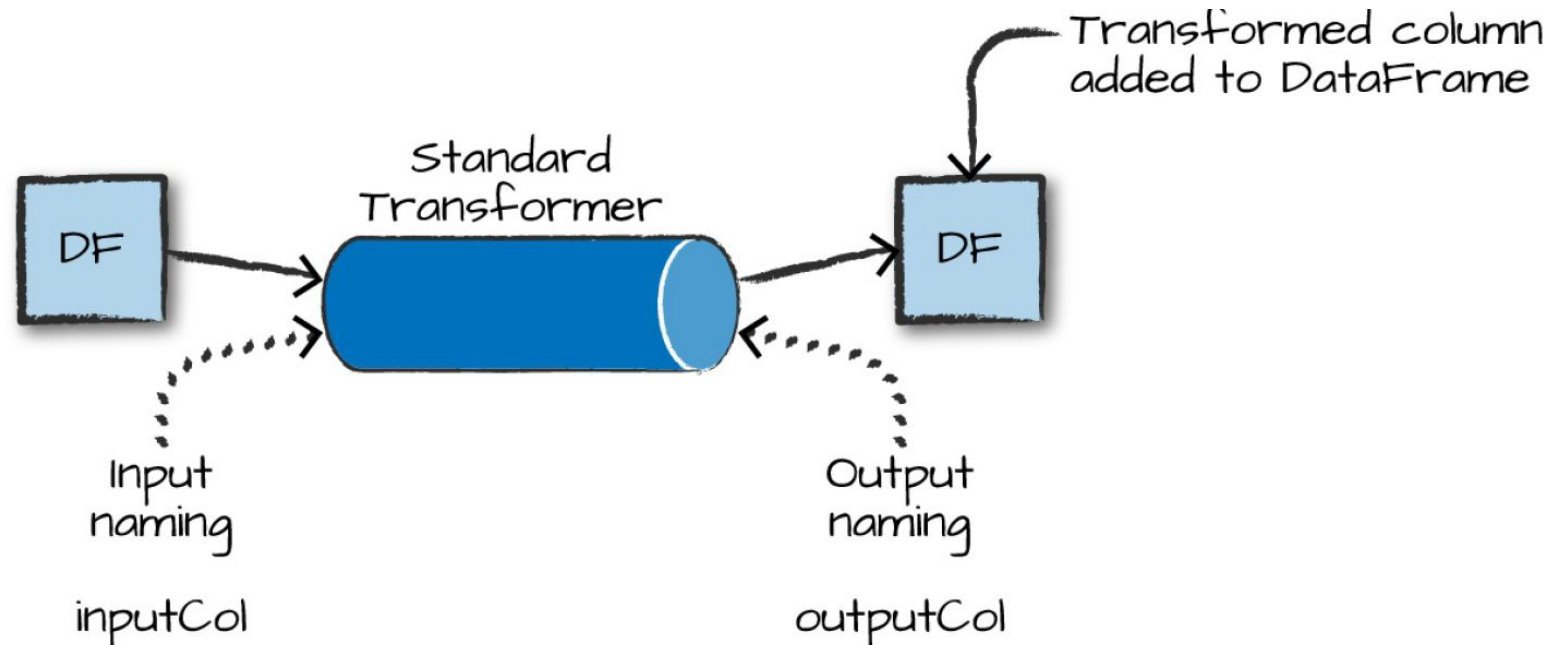
The machine learning workflow in Spark



Pipeline components. Transformer and Estimator

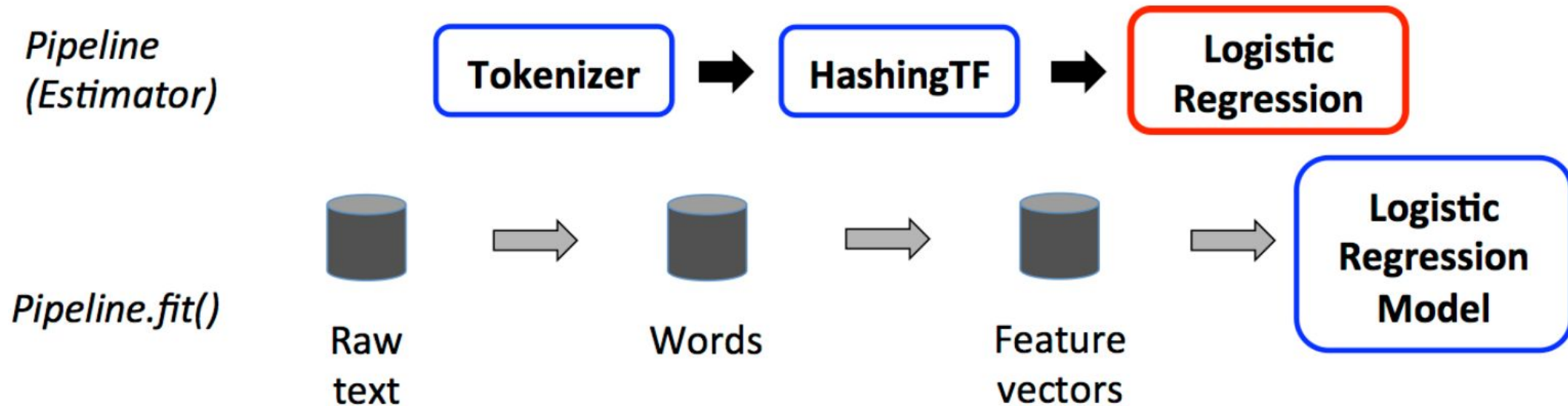
A Transformer is an abstraction that includes feature transformers and learned models

An Estimator abstracts the concept of a learning algorithm or any algorithm that fits or trains on data (est.fit(df) method)



A Pipeline.

Blue — transformations, red — estimator

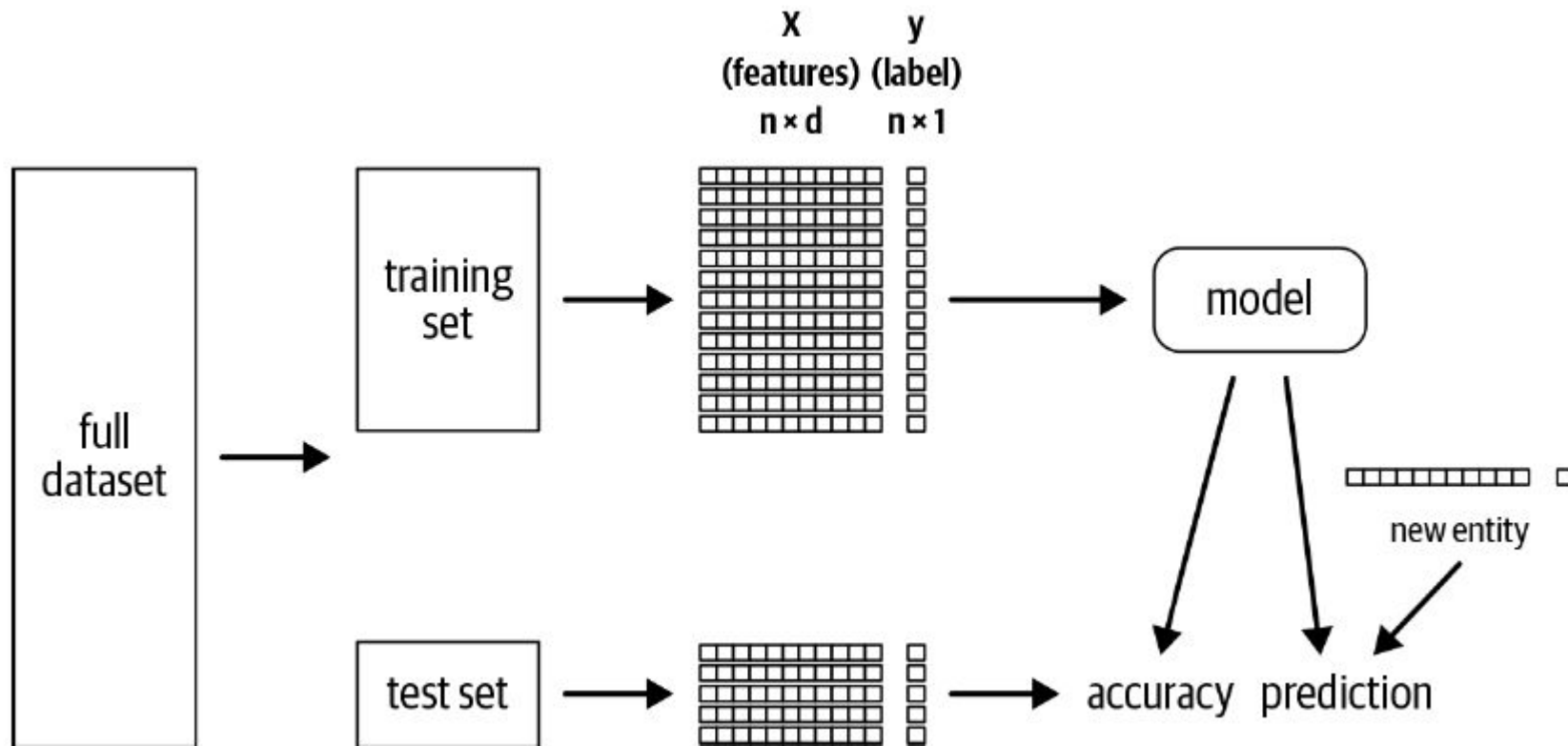


Sparse data types. Vector

```
1 from pyspark.ml.linalg import Vectors
2 denseVec = Vectors.dense(1.0, 2.0, 3.0)
3 size = 3
4 idx = [1, 2] # locations of non-zero elements in vector
5 values = [2.0, 3.0]
6 sparseVec = Vectors.sparse(size, idx, values)
```



Creating Training and Test Data Sets



Linear Regression

predictor, 'x-variable',
independent variable,
explanatory variable

coefficient

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

linear predictor

response, dependent variable,
observation, 'y-variable'

The diagram illustrates the components of a linear regression equation. The equation is $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$. The variable Y is circled in red, with a red arrow pointing to the text 'response, dependent variable, observation, 'y-variable''. The term x_1 is circled in green, with a green arrow pointing to the text 'predictor, 'x-variable', independent variable, explanatory variable'. The coefficient β_2 is circled in orange, with an orange arrow pointing to the text 'coefficient'. A blue bracket under the entire right-hand side of the equation (from β_0 to $\beta_p x_p$) is labeled 'linear predictor'.

LogisticRegression

```
1 from pyspark.mllib.linalg import Vectors
2 from pyspark.ml.classification import LogisticRegression
3 from pyspark.ml.param import Param, Params
4
5 # Prepare training data from a list of (label, features) tuples.
6 training = sqlContext.createDataFrame([
7     (1.0, Vectors.dense([0.0, 1.1, 0.1])),
8     (0.0, Vectors.dense([2.0, 1.0, -1.0])),
9     (0.0, Vectors.dense([2.0, 1.3, 1.0])),
10    (1.0, Vectors.dense([0.0, 1.2, -0.5]))], ["label", "features"])
11
12 # Create a LogisticRegression instance. This instance is an Estimator.
13 lr = LogisticRegression(maxIter=10, regParam=0.01)
14 # Print out the parameters, documentation, and any default values.
15 print "LogisticRegression parameters:\n" + lr.explainParams() + "\n"
16
17 # Learn a LogisticRegression model. This uses the parameters stored in
18    lr.
19 model1 = lr.fit(training)
```

Ваши вопросы?

BigData. Фреймворк Apache Spark, урок 4

