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.

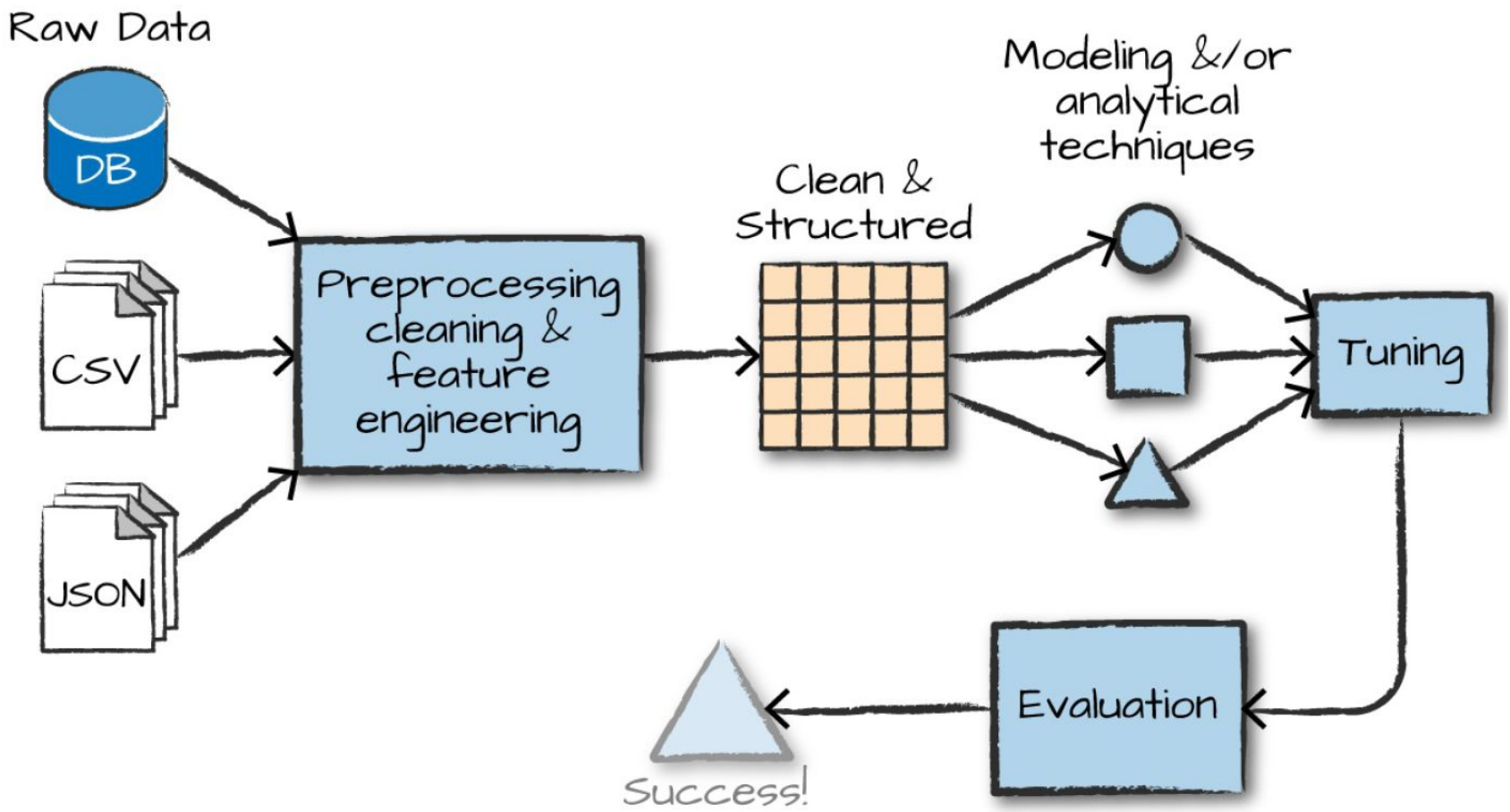
Ссылка

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

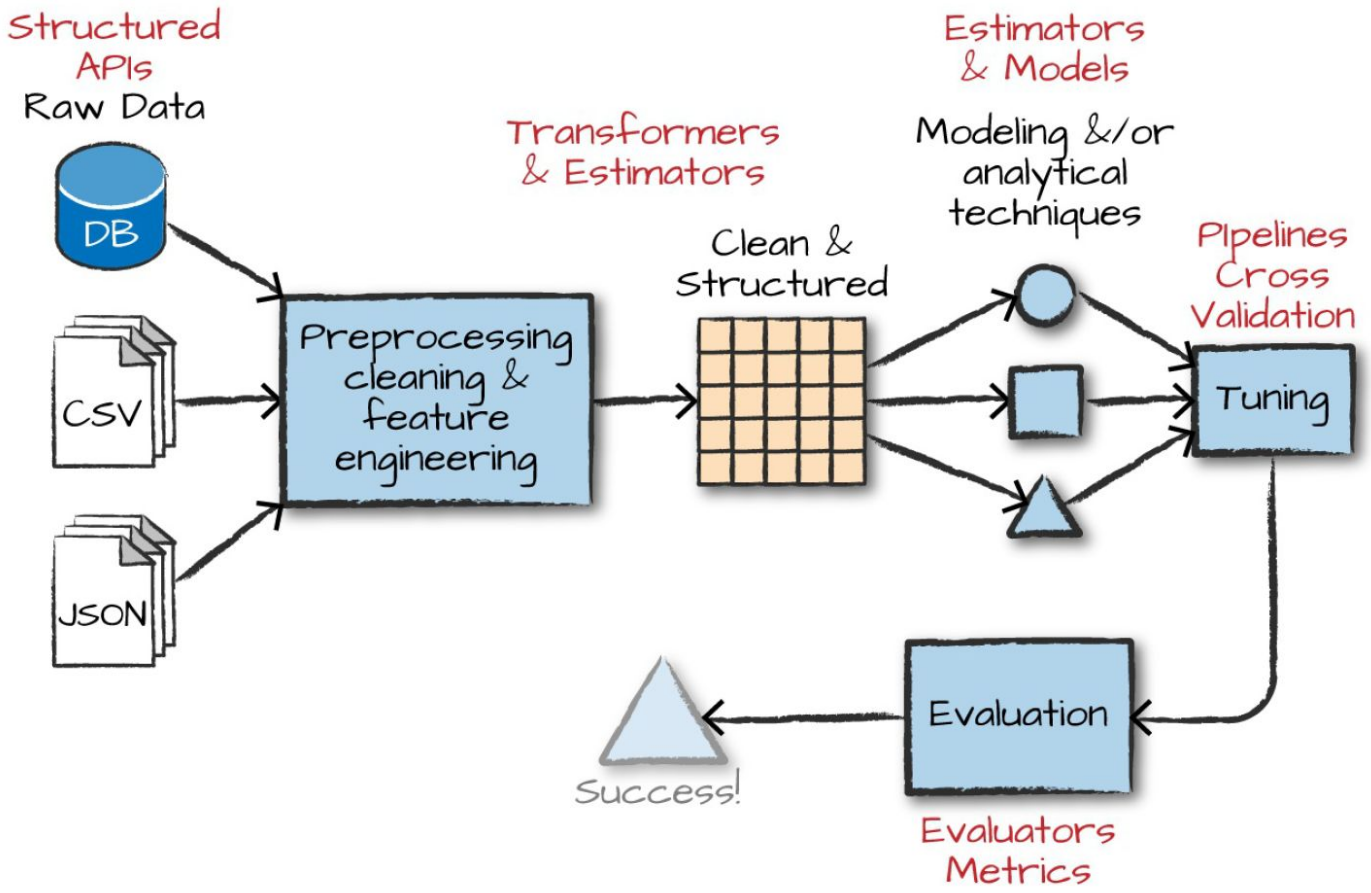
- ⚡ **Чистый код** за счёт автоматизации процедур подготовки данных — выборка, очистка, генерация предикторов (фичей, от англ. feature) и пр.
- ⚡ **Сокращение ошибок** благодаря отработанной последовательности шагов (пропустить или неправильно выполнить какой-то этап не получится)
- ⚡ **Простота развёртывания 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

Ссылка

The machine learning workflow



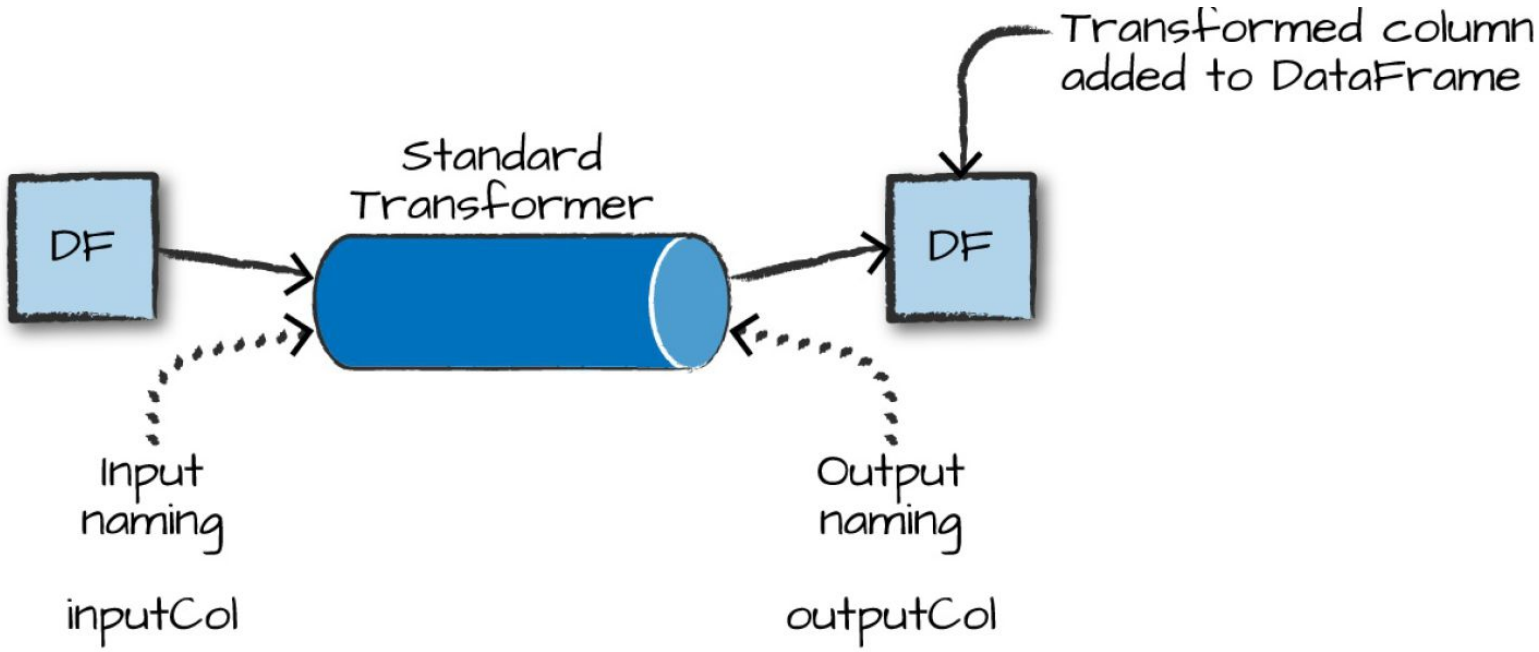
The machine learning workflow in Spark



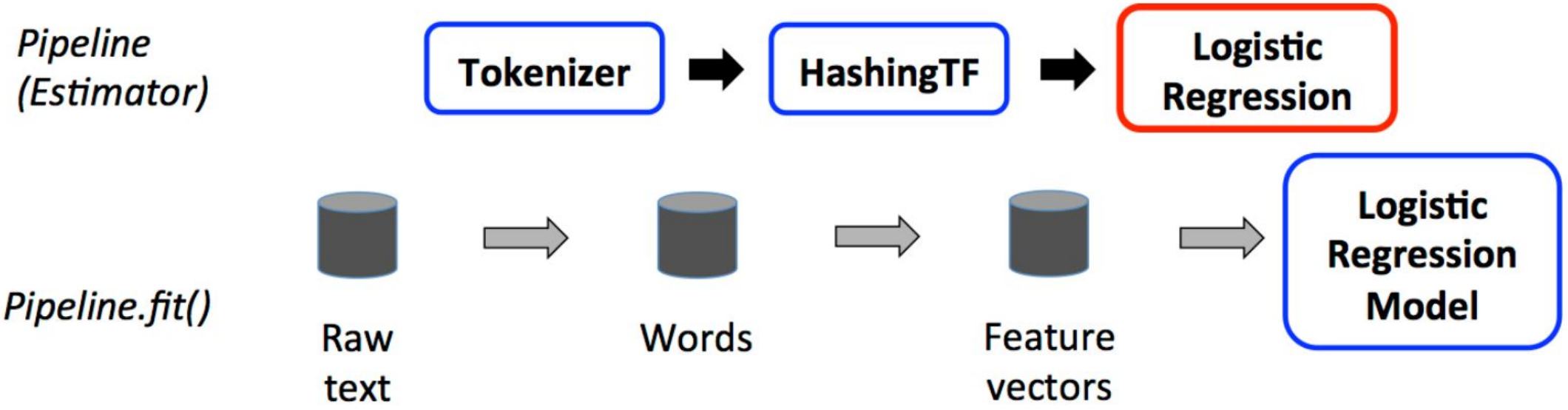
Pipeline components. Transformer and Estimator

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)



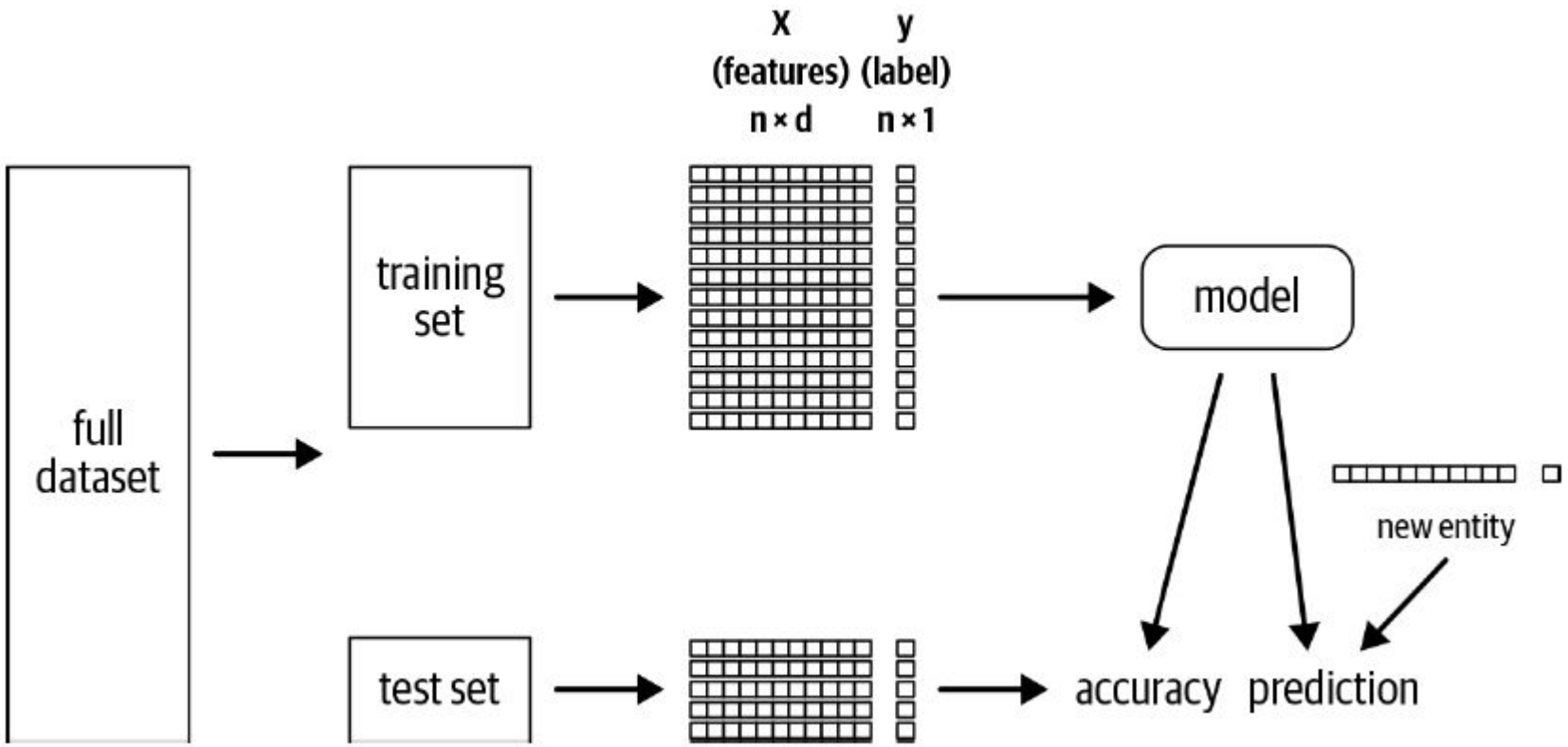
A Pipeline.
Blue — transformations, red — estimator



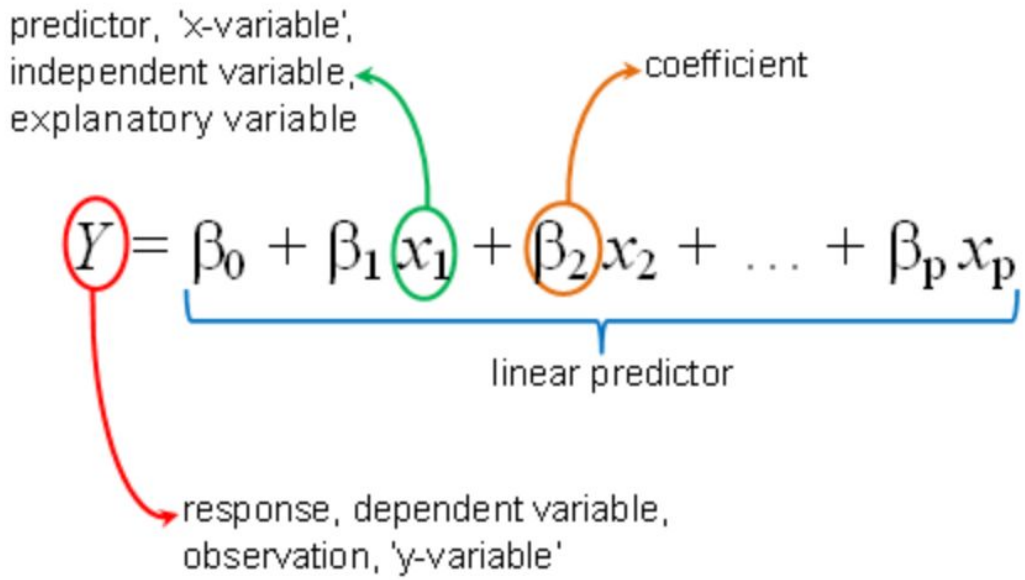
Sparse data types. Vector

```
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



Ссылка

LogisticRegression

```
from pyspark.mllib.linalg import Vectors
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.param import Param, Params

# Prepare training data from a list of (label, features) tuples.
training = sqlContext.createDataFrame([
    (1.0, Vectors.dense([0.0, 1.1, 0.1])),
    (0.0, Vectors.dense([2.0, 1.0, -1.0])),
    (0.0, Vectors.dense([2.0, 1.3, 1.0])),
    (1.0, Vectors.dense([0.0, 1.2, -0.5]))], ["label", "features"])

# Create a LogisticRegression instance. This instance is an Estimator.
lr = LogisticRegression(maxIter=10, regParam=0.01)
# Print out the parameters, documentation, and any default values.
print "LogisticRegression parameters:\n" + lr.explainParams() + "\n"

# Learn a LogisticRegression model. This uses the parameters stored in
lr.
model1 = lr.fit(training)
```

Ссылка

Ваши вопросы?

BigData. Фреймворк Apache Spark, урок 4