

# Распределенное обучение моделей **Spark ML**

# Что позволяет **Spark ML**?

<http://spark.apache.org/docs/latest/ml-features.html>

# Spark ML

**Spark ML (или MLlib)** — это библиотека машинного обучения в составе Apache Spark, предназначенная для масштабируемой обработки и построения моделей на больших данных.

# Основные особенности **Spark ML**

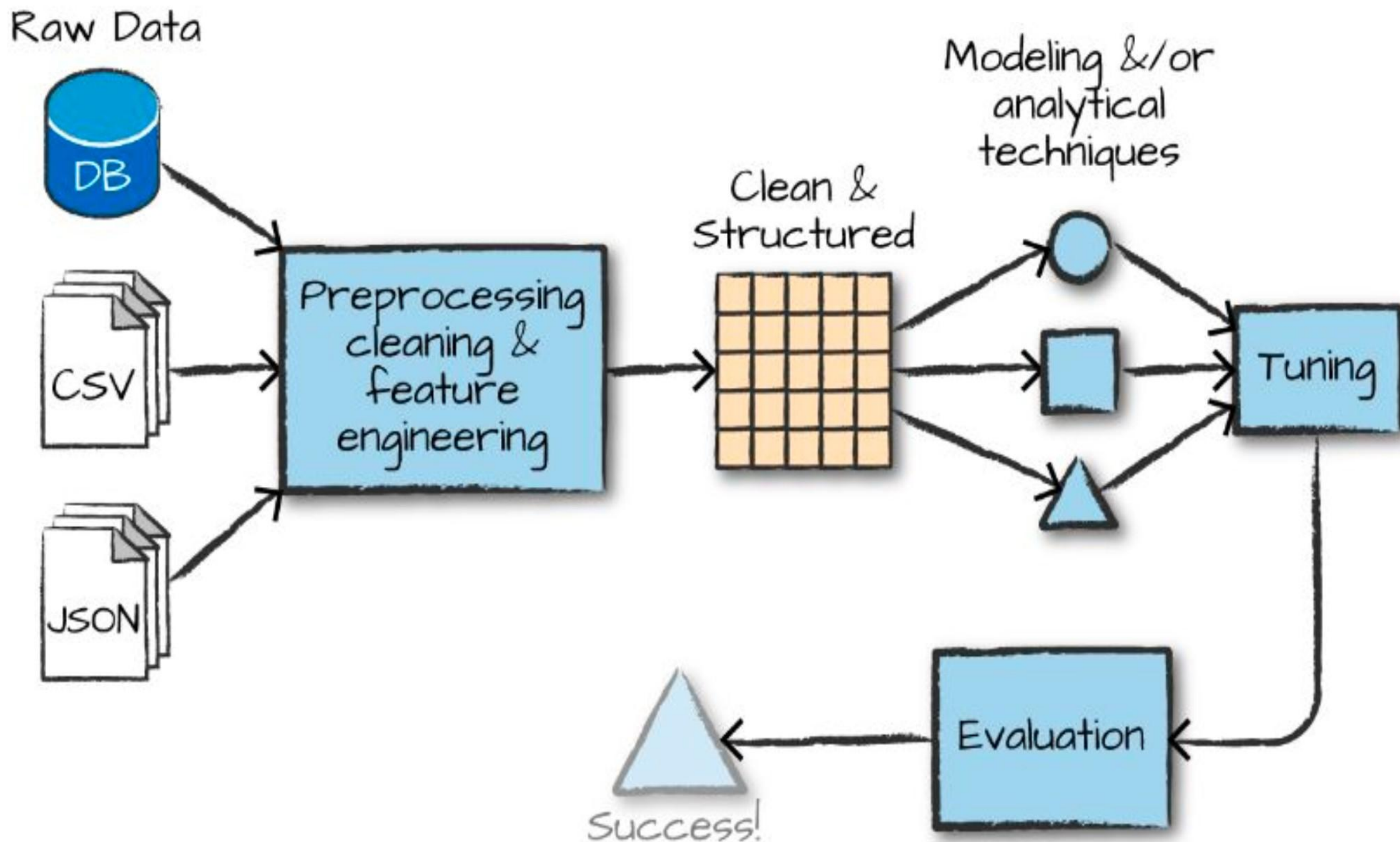
1. **Работает на больших данных** — может обрабатывать терабайты данных распределённо.
2. **Pipeline API** — позволяет строить цепочки обработки данных и моделей (аналог sklearn Pipelines).
3. Поддержка популярных алгоритмов:
  1. Классификация: Logistic Regression, Decision Trees, Random Forest, Gradient-Boosted Trees
  2. Регрессия
  3. Кластеризация (K-Means и др.)
  4. Поиск рекомендаций (ALS)
  5. Работа с текстами (TF-IDF, Word2Vec, Tokenizer и др.)
4. **Совместимость с DataFrame API**, что делает его удобным для предобработки данных в стиле SQL/Pandas.

# Модули **Spark ML**

`spark.mllib` – Старая API, основанная на RDD. Устаревает.

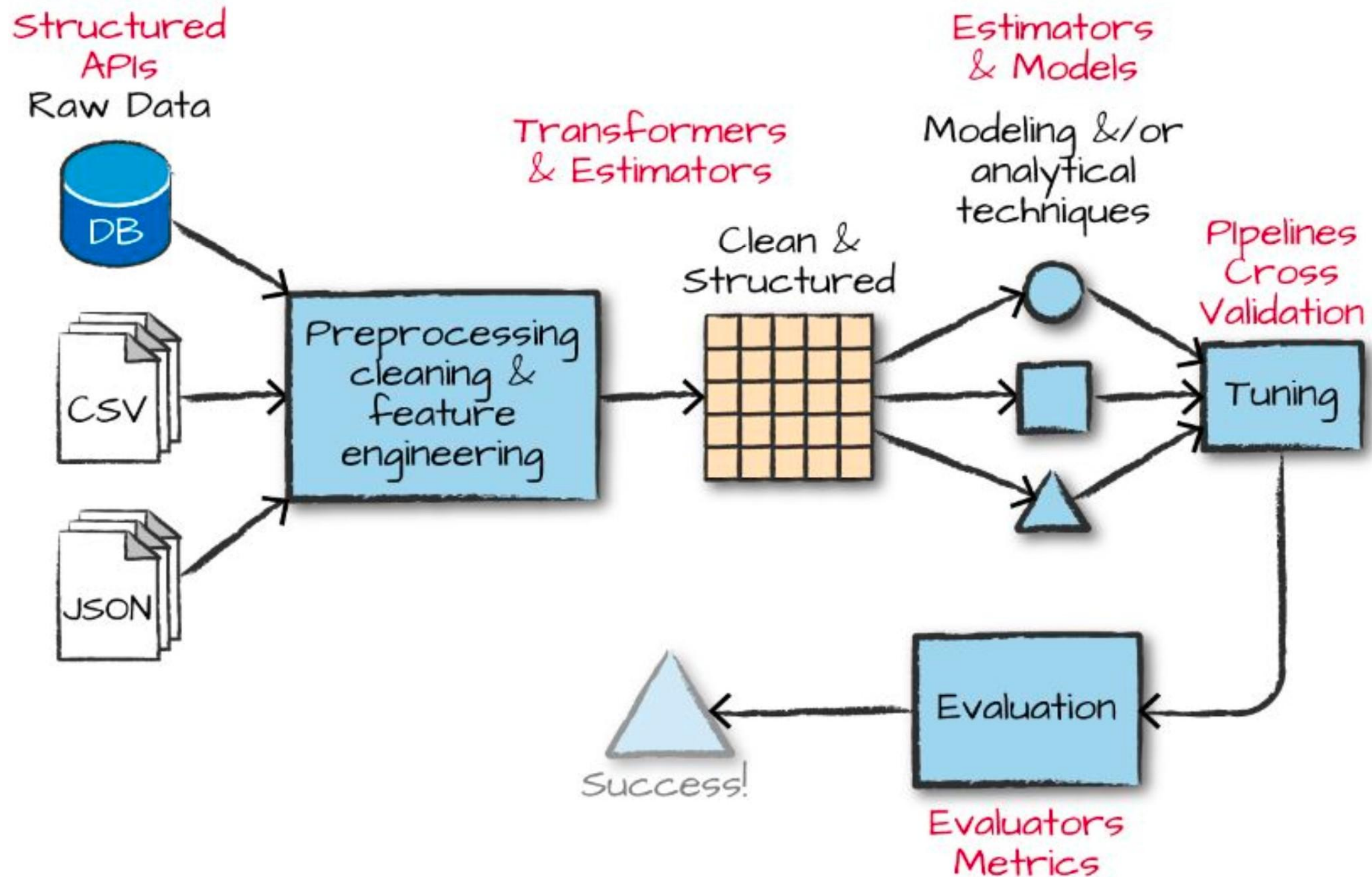
`spark.ml` – Новая API, основанная на DataFrame. Рекомендуется к использованию.

# Процесс построения модели машинного обучения





# Конвейер (pipeline) машинного обучения в Apache Spark ML



# Pipeline Spark ML

## 1. Raw Data (Сырые данные)      Источники данных:

1. Базы данных (DB)
2. CSV-файлы
3. JSON-файлы

## 2. Preprocessing, cleaning & feature engineering (Предобработка, очистка и создание признаков)      На этом этапе происходит:

1. Удаление пропусков
2. Очистка от шумов и выбросов
3. Преобразование категориальных признаков
4. Масштабирование числовых
5. Создание новых признаков (feature engineering)

## 3. Clean & Structured (Чистые и структурированные данные)

Результат предобработки — аккуратный DataFrame/таблица, готовая для подачи в модели.

## 4. Modeling and/or analytical techniques (Моделирование и/или аналитика)

Попробуем разные модели (круг, квадрат, треугольник — как абстрактные обозначения разных алгоритмов, например:

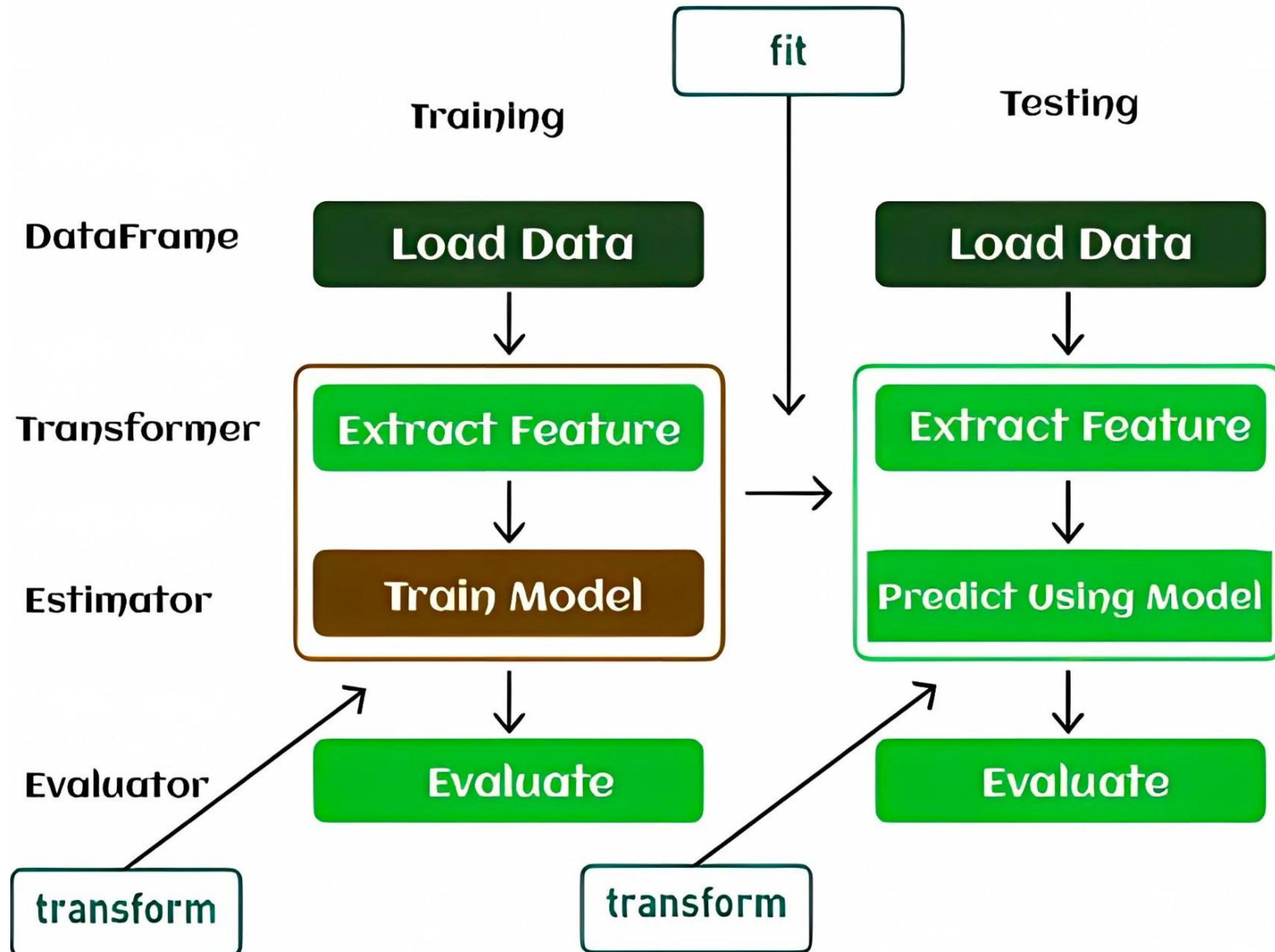
1. логистическая регрессия
2. дерево решений
3. случайный лес
4. кластеризация и пр.)

## 5. Tuning (Настройка гиперпараметров)

Автоматический или ручной подбор параметров моделей для повышения точности.

Примеры: GridSearch, RandomSearch, Cross-validation.





# Процесс построения и применения модели

## Обучение модели:

1. **Load Data**  
Загрузка обучающего набора данных в формате `Spark DataFrame`.
2. **Extract Feature (*Transformer*)**  
Преобразование исходных данных в набор признаков (features):
  1. `Tokenizer`, `VectorAssembler`, `StringIndexer` и др.
  2. Это этап `transform()`
3. **Train Model (*Estimator*)**  
Обучение модели на признаках:
  1. `LogisticRegression().fit(data)`
  2. После этого `Estimator` становится `Model`, т.е. `Transformer`.
4. **Evaluate (*Evaluator*)**  
Оценка качества модели на обучающей выборке по метрикам (`accuracy`, `f1`, `rmse`, и т.д.)

## Тестирование модели:

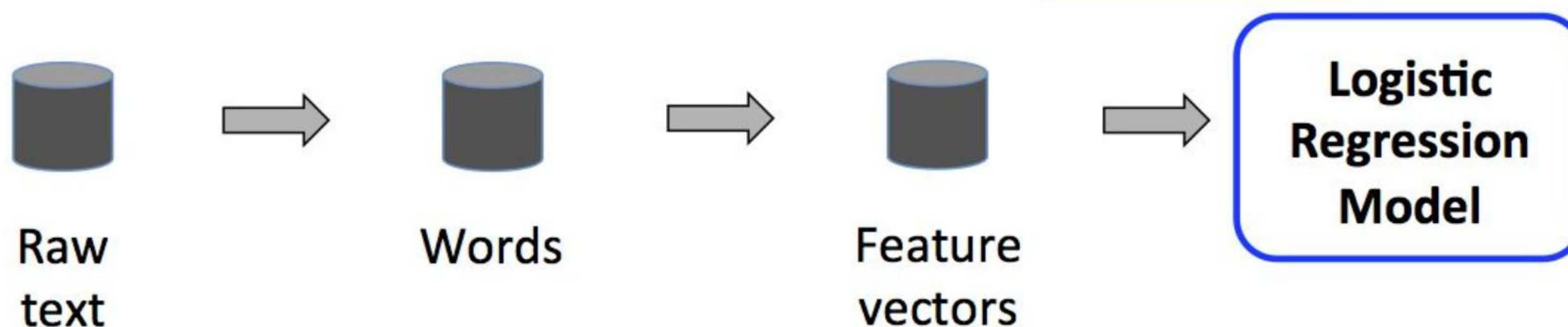
1. **Load Data**  
Загрузка тестового набора данных (который не использовался для обучения).
2. **Extract Feature (*Transformer*)**  
Применение тех же преобразований признаков (важно: **тот же пайплайн**).
3. **Predict Using Model (*Transformer*)**  
Использование обученной модели для предсказания:  
`predictions = model.transform(testData)`
4. **Evaluate (*Evaluator*)**  
Оценка качества на тестовой выборке.

# Pipeline

*Pipeline  
(Estimator)*



*Pipeline.fit()*



*PipelineModel  
(Transformer)*



*PipelineModel  
.transform()*





# Transformers

- Tokenizer
- StopWordsRemover
- n-gram
- Binarizer
- PCA
- PolynomialExpansion
- Discrete Cosine Transform (DCT)
- StringIndexer
- IndexToString
- OneHotEncoder
- VectorIndexer
- Interaction

- Normalizer
- StandardScaler
- RobustScaler
- MinMaxScaler
- MaxAbsScaler
- Bucketizer
- ElementwiseProduct
- SQLTransformer
- VectorAssembler
- VectorSizeHint
- QuantileDiscretizer
- Imputer

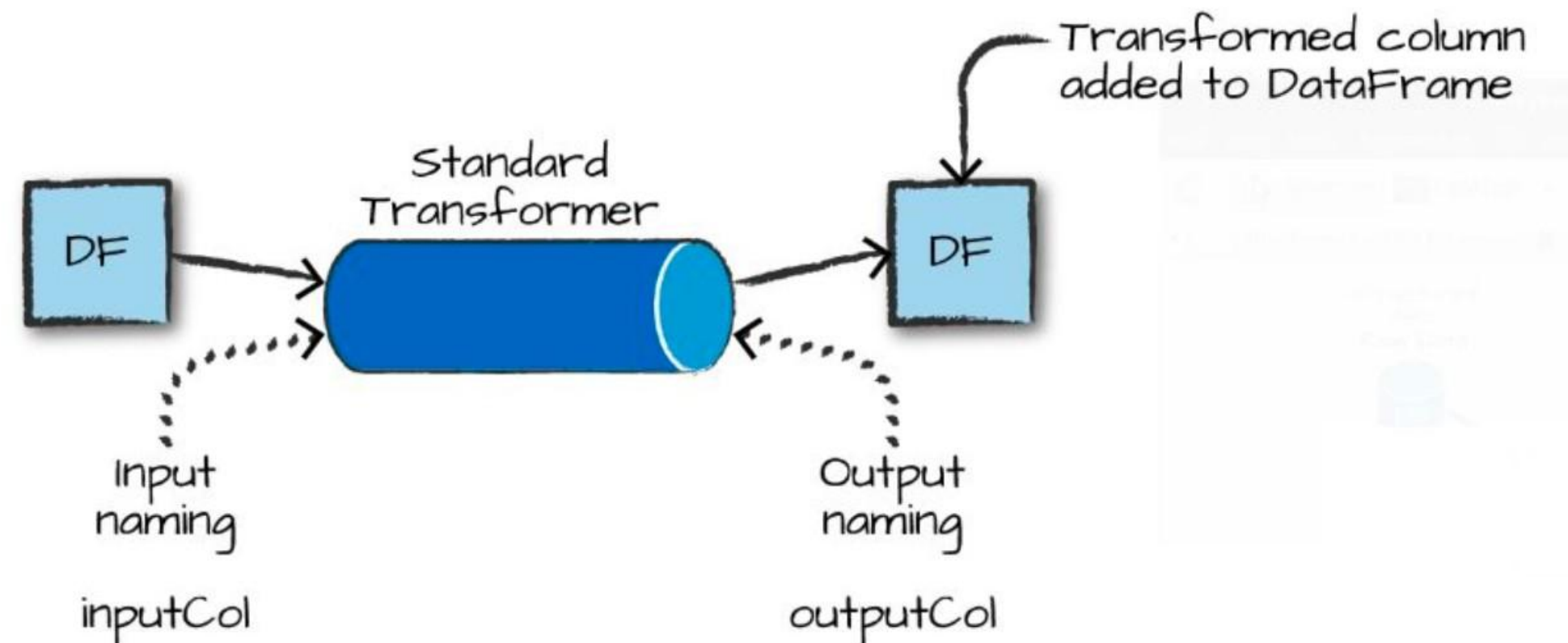


Figure 24-3. A standard transformer

# Estimators

- LogisticRegression
- DecisionTreeClassifier
- RandomForestClassifier
- NaiveBayes
- OneVsRest
- MultilayerPerceptronClassifier

- KMeans
- LDA
- GaussianMixture
- ALS
- DecisionTreeRegressor
- LinearRegression
- RandomForestRegressor

# Evaluators

- `BinaryClassificationEvaluator`
- `RegressionEvaluator`
- `MulticlassClassificationEvaluator`
- `MultilabelClassificationEvaluator`
- `ClusteringEvaluator`



**Спасибо за внимание!**