Lab 3 – Machine Learning

Spark Capabilities

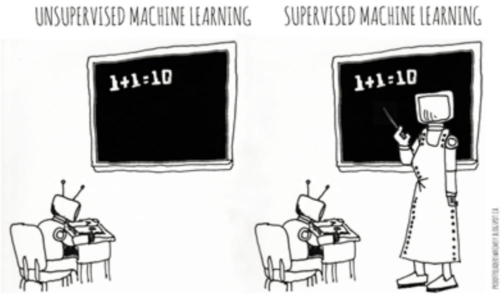
Spark Core

Spark Streaming	**Stream Processing** Near real-time data processing & analytics	• **Micro-batch event processing** for near real-time analytics • Process live streams of data (IoT, Twitter, Kafka) • No multi-threading or parallel processing required
MLlib (machine learning)	**Machine Learning** Incredibly fast, easy to deploy algorithms	• **Predictive and prescriptive analytics**, and smart application design, from statistical and algorithmic models • Algorithms are pre-built
Spark SQL	**Unified Data Access** Fast, familiar query language for all data	• **Query your structured data sets** with SQL or other dataframe APIs • Data mining, BI, and insight discovery • Get results faster due to performance
GraphX (graph)	**Graph Analytics** Fast and integrated graph computation	• **Represent data in a graph** • Represent/analyze systems represented by nodes and interconnections between them • Transportation, person to person relationships, etc.

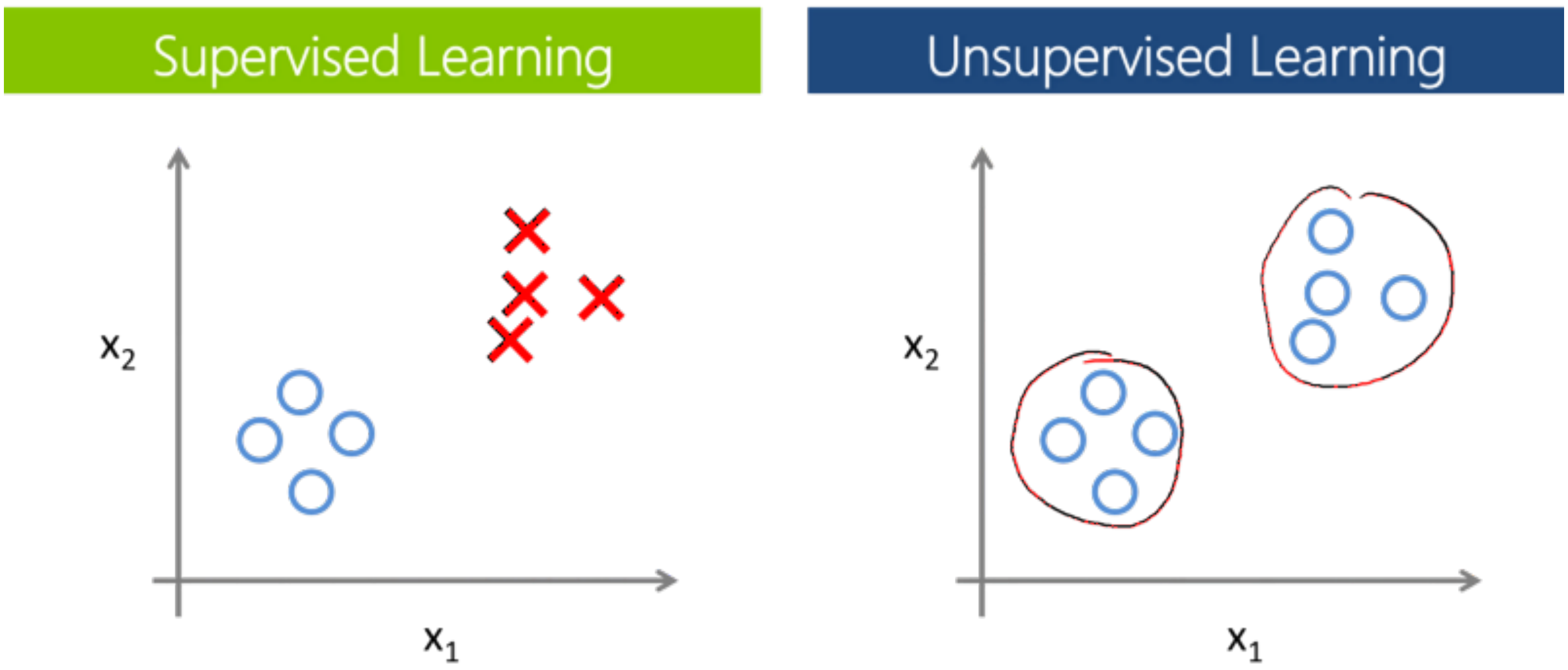
Categories of Machine Learning

- **Supervised learning**
  - The program is “trained” on a pre-defined set of “training examples”, which then facilitate its ability to reach an accurate conclusion when given new data
  - The algorithm is presented with example inputs and their desired outputs (correct results)
  - The goal is to learn a general rule that maps inputs to outputs

- **Unsupervised learning**
  - No labels are given to the learning algorithm, leaving it on its own to find structure (patterns and relationships) in its input
  - Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning)

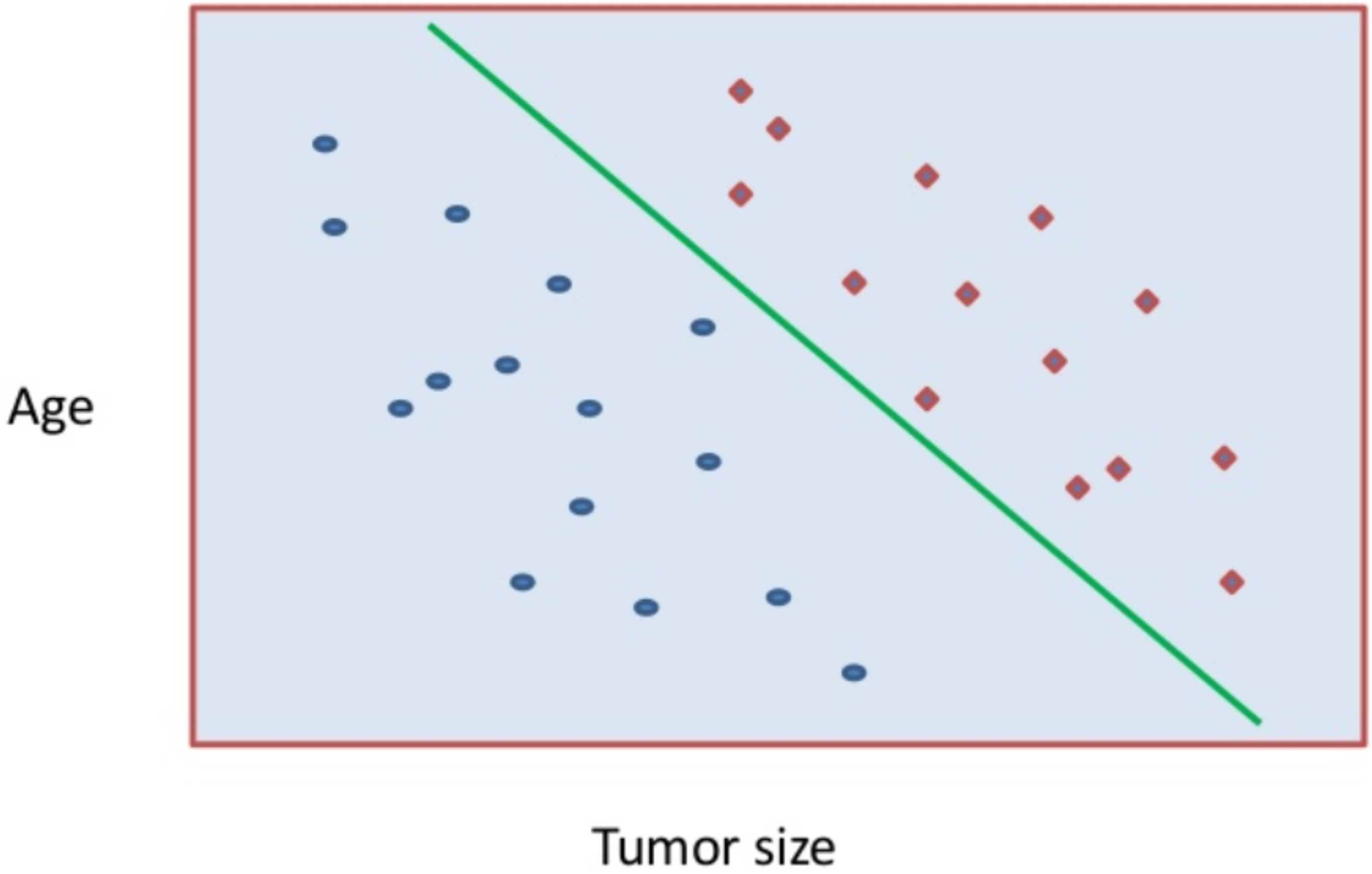


Supervised vs. Unsupervised Learning



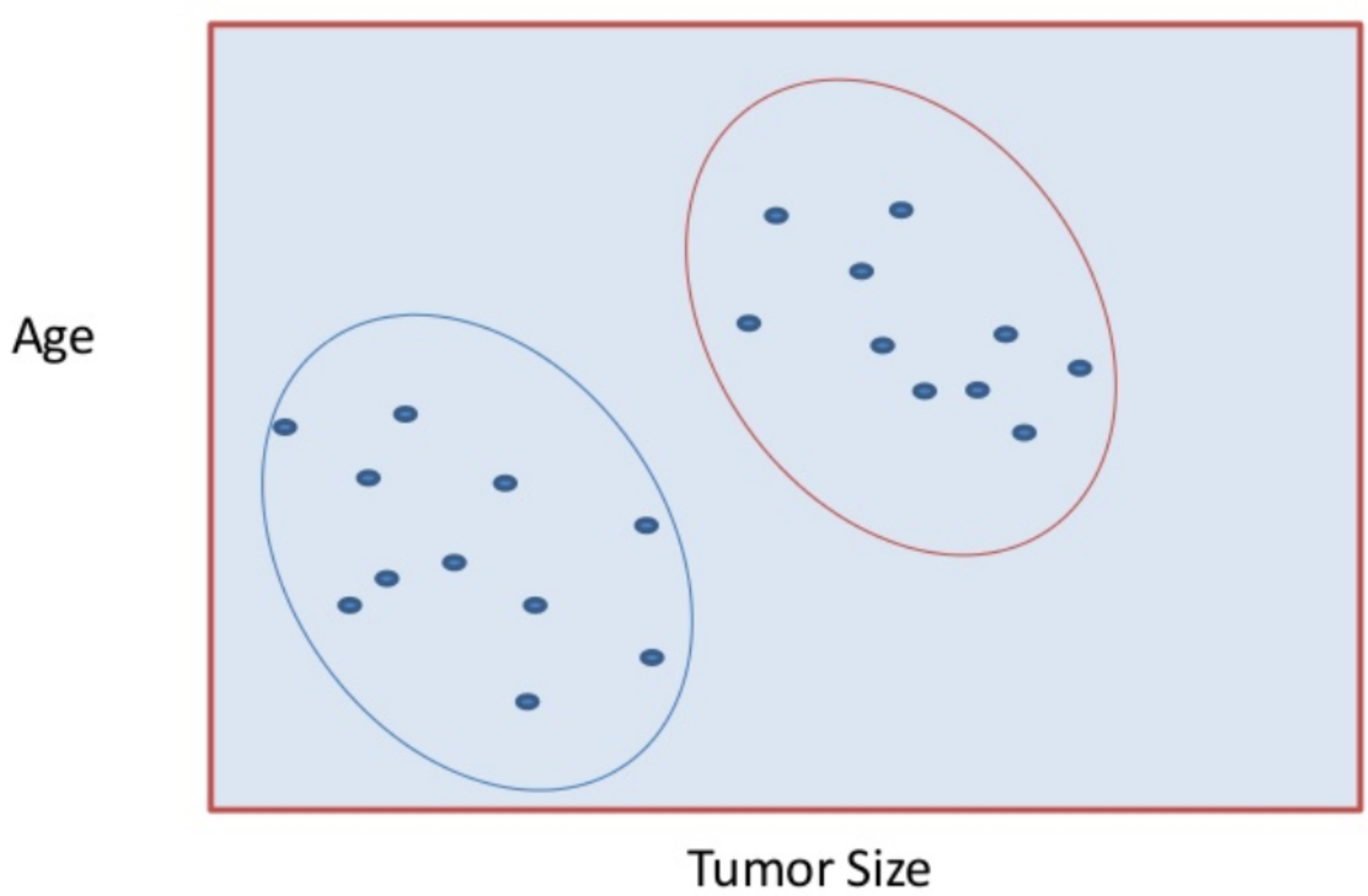
Example of Supervised Learning (Classification)

Goal is to make predictions

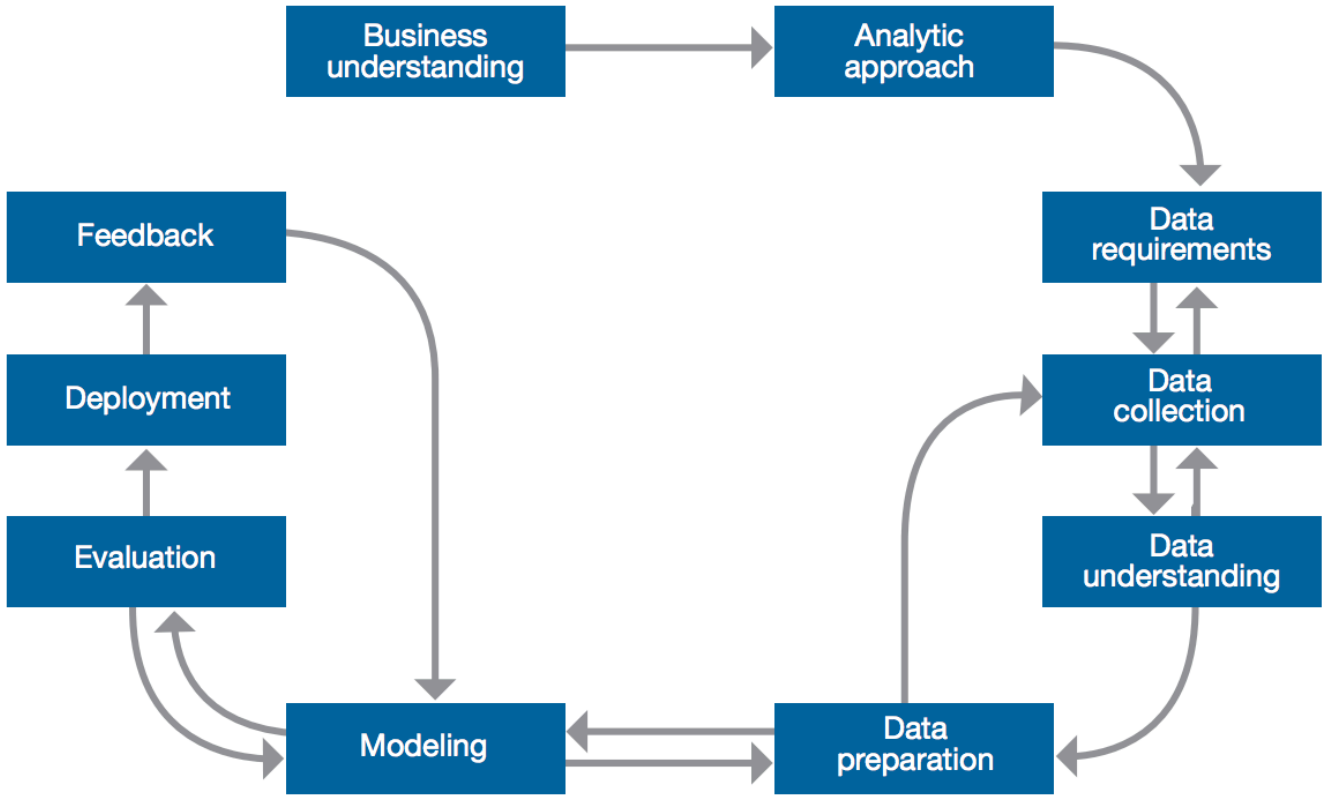


Example of Unsupervised Learning (Clustering)

Goal is to understand the structure of the data, not make predictions



Typical Machine Learning Process and Pipeline



Beyond just the algorithms,, successful implementation of machine learning projects requires a process and rigor to achieve a useful result.

Spark ML

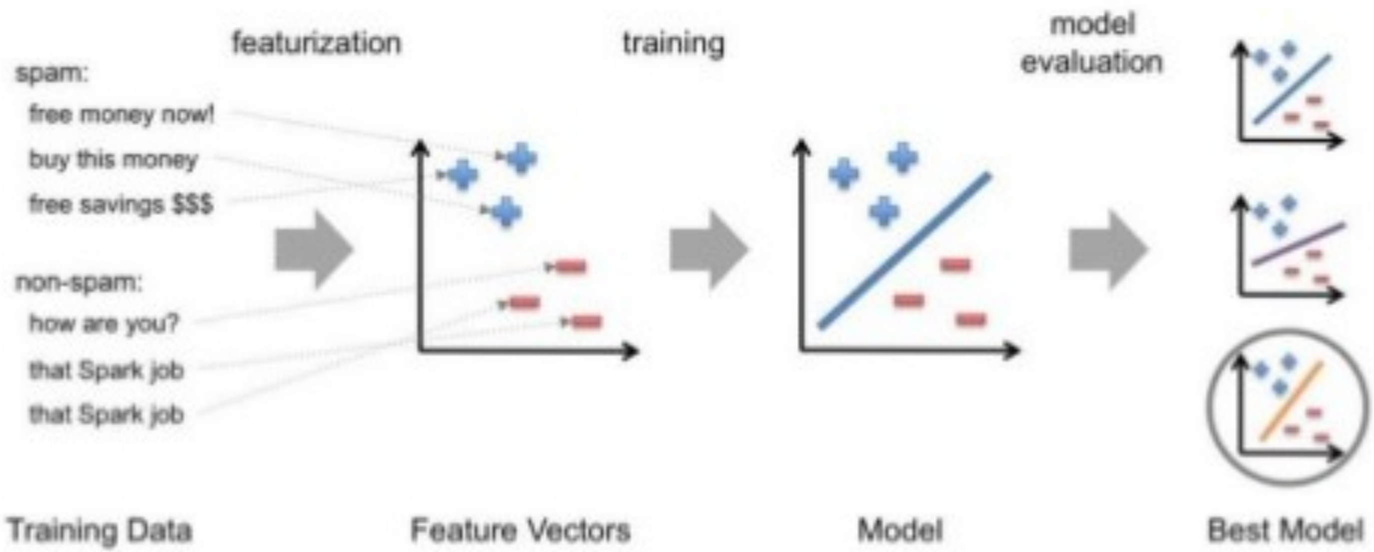
- Spark ML is Spark's machine learning (ML) library
- Its goal is to make practical machine learning scalable and easy
- Consists of common learning algorithms and utilities, including
  - Classification
  - Regression
  - Clustering
  - Collaborative filtering
  - Dimensionality Reduction
- Lower-level optimization primitives
- Higher-level pipeline APIs

Spark ML

- Divides into two packages:
  – spark.mllib contains the original API built on top of RDDs
  – spark.ml provides higher-level API built on top of DataFrames for constructing ML pipelines

- Using spark.ml is recommended because with DataFrames the API is more versatile and flexible
  – spark.mllib will continue to be supported

Typical Steps in ML Pipeline



Recommendation Systems

- **Recommendation systems seek to predict the rating (or preference) that a user would give to an item**

- **Recommendation systems attempt to improve customer experience through personalized recommendations based on prior user feedback**

- **Recommender systems have become extremely common in recent years, and are applied in a variety of applications**
  - movies, music, news, books, research articles, search queries, social tags, …
  - products in general

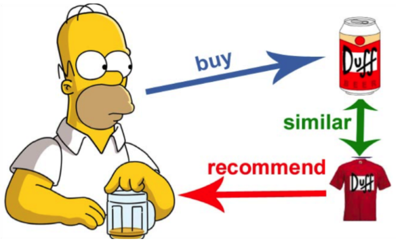
- **Collaborative filtering is a technique that is commonly used for recommender systems**
  - employs a form of wisdom of the crowd approach

Collaborative Filtering with Spark ML

- **Forms of Collaborative Filtering**
  – Explicit matrix factorization - preferences provided by users themselves are utilized
  – Implicit matrix factorization - only implicit feedback (e.g. views, clicks, purchases, likes, shares etc.) is utilized

- **Spark ML supports an implementation of matrix factorization for collaborative filtering**
  – Matrix factorization models have consistently shown to perform extremely well for collaborative filtering

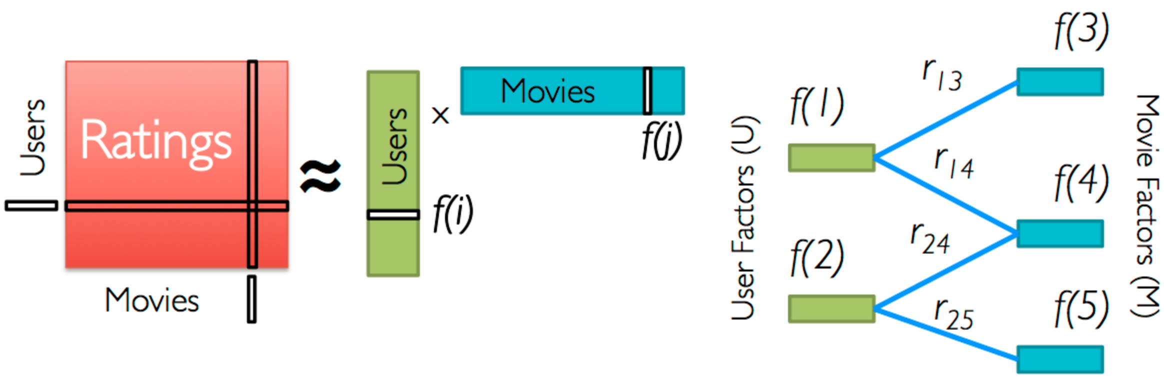
- **Collaborative filtering aims to fill in the missing entries of a user-item association matrix in which users and items are described by a small set of latent factors that can be used to predict missing entries**



Alternating Least Squares (ALS) Algorithm

- **Spark ML uses the Alternating Least Squares (ALS) algorithm to learn the latent factors for the matrix factorization problem**
- **ALS works by iteratively solving a series of least square regression problems to derive a model**

Low-Rank Matrix Factorization:

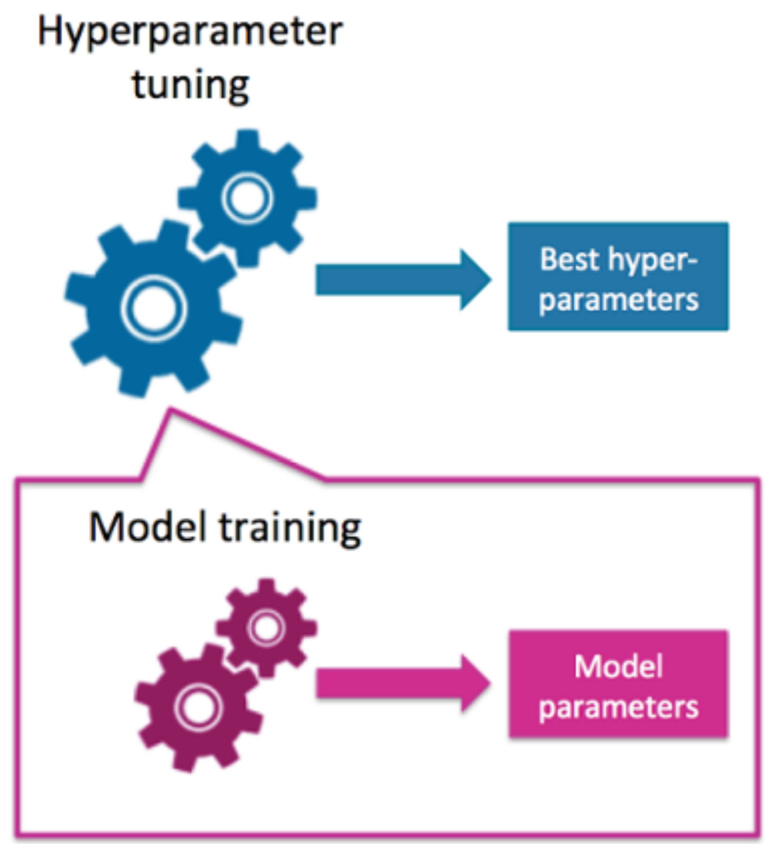


Iterate:

$$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} \left(r_{ij} - w^T f[j]\right)^2 + \lambda ||w||_2^2$$

Tuning a Spark ML Model - Hyper parameters

- Spark ML algorithms provide many hyperparameters for tuning models
- These hyperparameters are distinct from the model parameters being optimized by ML itself
- Hyperparameter tuning is accomplished by choosing the best set of parameters based on model performance on test data that the model was not trained with



Lab 3 Flow

1. Download compressed CSV data and load into an RDD

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/10 8:26	2.55	17850	United Kingdom
536365	71053	WHITE METAL LANTERN	6	12/1/10 8:26	3.39	17850	United Kingdom
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/10 8:26	2.75	17850	United Kingdom

2. Prepare the data

– Remove header
– Only keep rows that have
  • a purchase quantity greater than 0
  • a non blank customer ID
  • a non blank stock code after removing non-numeric characters

4. Create a DataFrame from the resulting RDD

– Add a label column

5. Split the dataset

– 80% for training
– 10% for testing
– 10% for cross validation

Lab 3 Flow (continued)

5. **Build a recommendation model using the training dataset**
   – Two models using different hyperparameters
     - rank
     - maxIter

6. **Test the two models using the cross validation dataset**

7. **Evaluate the two models using mean squared error**
   – Confirm "best" model against the test dataset

8. **Use the "best" model to make predictions for a particular user**
   – Top 5 recommendations

```
                         description
0      YELLOW FLOWERS FELT HANDBAG KIT
1  MIDNIGHT BLUE COPPER FLOWER NECKLAC
2                   TEA TIME TEA TOWELS
3           BLACK DROP CRYSTAL NECKLACE
4   COPPER/OLIVE GREEN FLOWER NECKLACE
```