# Machine Learning At Scale

**Clustering**

Clustering is partitioning data points into related groups called clusters. In K means clustering, we make k such clusters.

A distance metric is used to group similar points together. Euclidean, Manhattan and Cosine distances can be used.

The K Means algorithm is

1. Select K points at random as cluster centers
2. For each data point, assign it to the closest cluster
3. For each cluster, recompute the center by averaging over all x axis points and all y axis points in the cluster
4. If the new centers are different from the old, repeat from step 2.

**K Means in MapReduce**

Map

1. Reads 2 files as input
2. Each map reads K centroids and one block from the dataset
3. Assign each point to the closest centroid
4. Output <centroid, point> where centroid is the key

Reduce

1. Get all points for a given centroid
2. Recompute new centroid
3. If new centroids same as old centroids or if iterations reached, stop
4. Else start another map-reduce job
5. Output <new centroid>

Some optimizations

1. Use a combiner - Computes each centroid as the local sum of assigned points and sends the centroid, partial sums
2. Use of single reducer
    a. Amount of data to reducers is very small, so reduce can compute the change in centers, creating a single output file.

**Collaborative filtering using Alternating Least Squares**

1. Express a User-Item Rating matrix R as a product of User vector A and Item vector B
2. Calculate A,B such that R ~ AB.
    a. Start with a random A and B
    b. On the $i^{th}$ iteration
        i. Assume $B_{i-1}$ is correct, calculate best value of $A_i$.
        ii. Assume $A_i$ is correct, calculate best value of $B_i$.
        iii. Loop till convergence

The best value can be calculated by taking the error as $R - A_i B_{i-1}^T$, and this can be minimised to find the $A_i$ that minimizes it. This comes out as

$A = (B_{i-1}^T B_{i-1}) B_{i-1}^T R^T$

**Spark MLLib**

The main disadvantages of ML algorithms at scale are

1. Creating and handling many RDDs of many types
2. Whole pipeline needs to be scripted and it is not modular
3. Parameter tuning takes overhead

ML Pipelines in Spark allow developers to just

1. Program to extract features
2. Specify the model used

This automates the process of writing a script, training the model, evaluating the model and deploying it in production.

The key concepts of Spark MLLib are

1. Provides an RDD abstraction of a dataframe
2. Introduces a streamlined ML pipeline
    a. Transformers
    b. Estimators
    c. Evaluators
3. Parameter tuning
    a. API
    b. Tuning

A data frame is a distributed collection of data organized into named columns, like a table. Its key features are

1. Scales up
2. Supports wide varieties of data formats
3. State of the art optimization and code generation
4. APIs in Python and Java

A data frame thus is an RDD + schema + Domain-Specific Language

Dataframes can be created by

1. Existing RDDs
2. Hive Tables
3. Data sources such as JSON

The ML Pipeline is given by

1. Transformers
    a. Extract features from Dataframes
    b. Features stored in new Dataframes
2. Estimators
    a. ML algorithms
    b. Can be user defined
3. Evaluators
    a. Compute predictions and metrics
    b. Tune algorithm parameters
    c. Depends on estimators