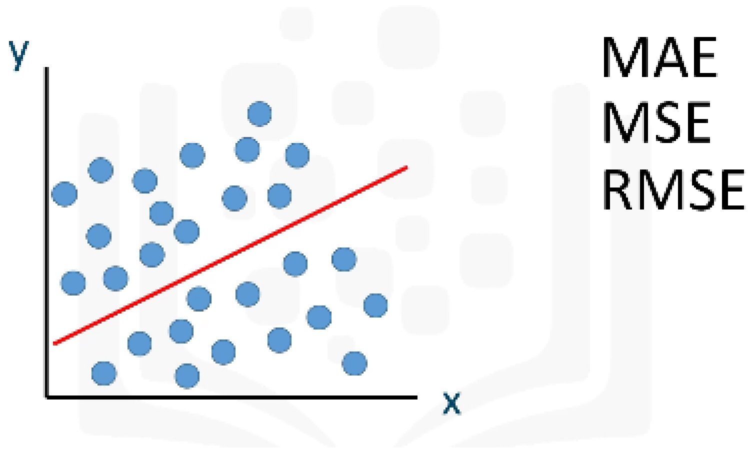
Lesson Objectives

- Review the algorithms that will be focused on in this course
  - Linear Regression
  - Support Vector Machines (SVM)
  - Logistic Regression
  - Decision Trees
  - Random Forests
  - K-Means Clustering
  - Gaussian Mixture Clustering
- Determine which algorithms require more review before proceeding

Linear Regression



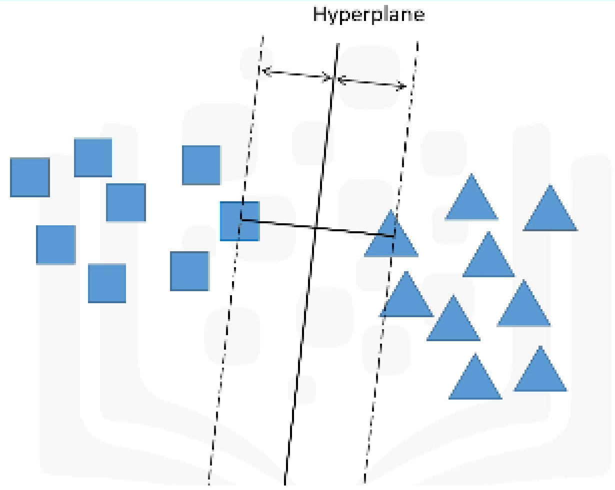
Evaluation Methods:

MAE = Mean Average Error

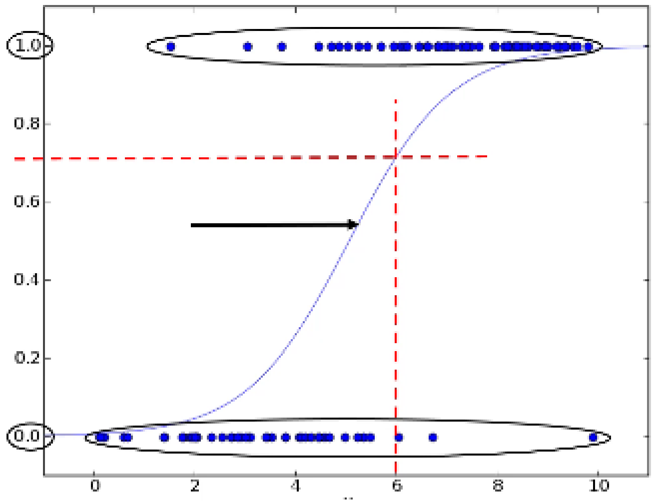
MSE = Mean Square Error

RMSE = Root Mean Square Error

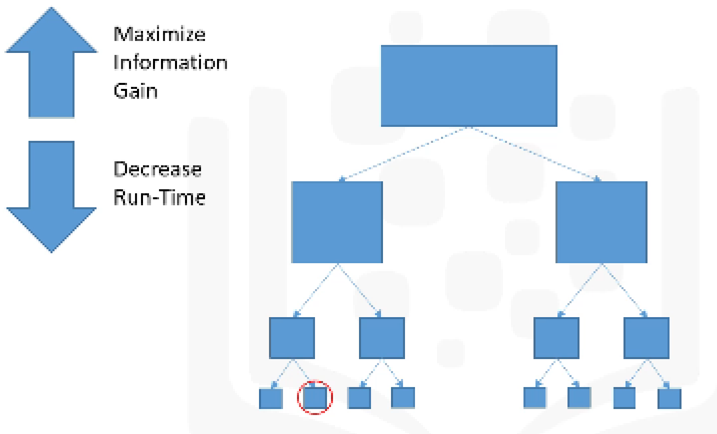
Support Vector Machines (SVM)



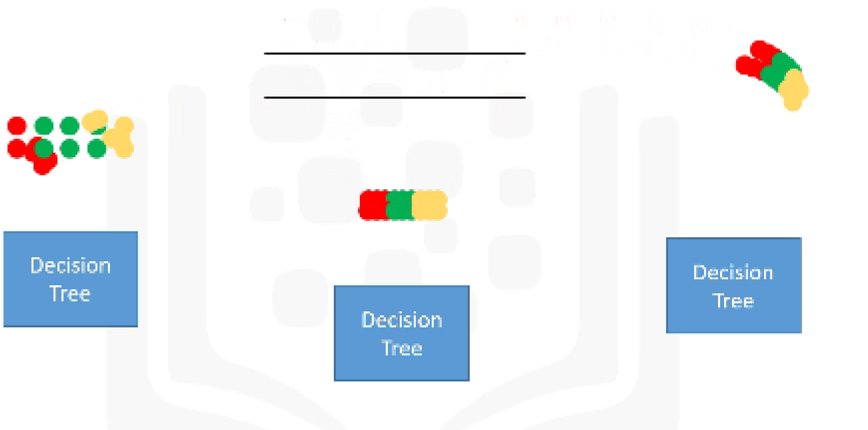
Logistic Regression



Decision Trees

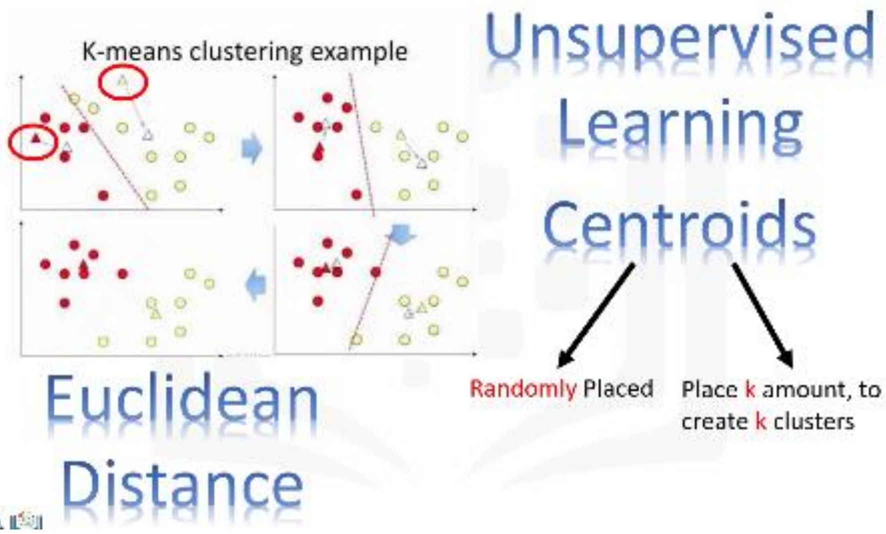


Random Forests

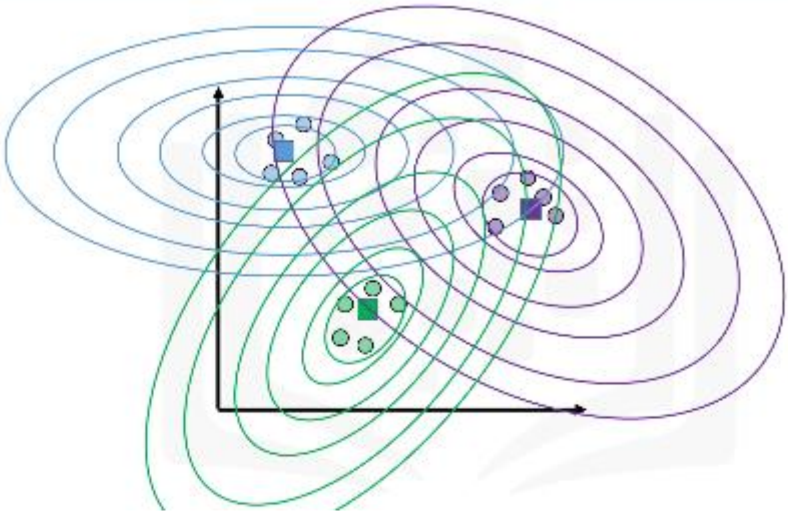


Train many decision trees. Put Input to many trees and average output

K-Means Clustering



Gaussian Mixture Clustering



Decision Trees & Split Candidates

Spark Mllib Supports:

- Binary Classification
- Multi-class Classification
- Regression Decision

Split Candidates

Continuous Features
&
Categorical Features

Split Candidates

Continuous Features

Small Datasets

Splits occur on unique values for the feature

Sort feature values, then use the ordered unique values as split candidates
Faster tree calculations

Large Datasets

Create sets of split candidates through quantile calculations on sample of the data

Splits create "bins". Maximum number of bins is specified with the maxBins parameter

Maximum number of bins cannot exceed the number of instances

Split Candidates

Categorical Features

For multiclass classification, use whenever possible

M categories → $(2^{M-1} - 1)$ possible split candidates

Binary Classification/Regression

$(M - 1)$ possible split candidates

If $(2^{M-1} - 1) > maxBins$

Similar to Binary Classification/Regression

Stopping Rule

Recursion stops if any of the following conditions are met.

- 1. Node depth is equal to the maxDepth training parameter
- 2. No split candidate leads to information gain greater than minInfoGain
- 3. No split candidate produces child nodes which have at least minInstancesPerNode training instances.

Lesson Objectives

- Investigate the different types of parameters involved in creating Decision Trees.
  - Specifiable Parameters (without Tuning required)
  - Tunable Parameters
  - Stopping Parameters (Tunable)
- Understand how varying the value used in parameters can affect the model positively or negatively

Decision Tree

Spark Mllib Supports:

- Binary Classification
- Multi-class Classification
- Regression Decision

Split Candidates

Continuous Features
&
Categorical Features

DecisionTree Parameters

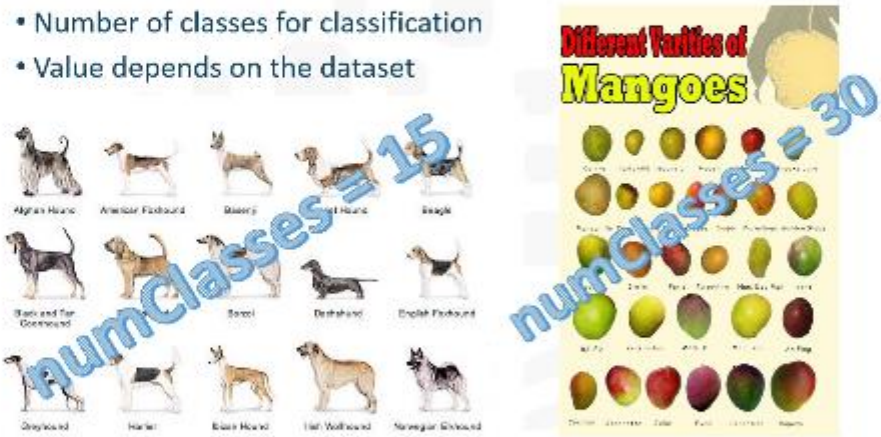
Specifiable Parameters (without Tuning required)
- numClasses, categoricalFeaturesInfo

Tunable Parameters
- maxBins, impurity

Stopping Parameters (Tunable)
- maxDepth, minInstancesPerNode, minInfoGain

Specifiable Parameters (No Tuning Required)

numClasses:
- Number of classes for classification
- Value depends on the dataset



Specifiable Parameters (No Tuning Required)

categoricalFeaturesInfo:
- Which features are categorical
- Number of values each feature can take
- Map from feature indices to number of categories.

Map(7 -> 6)

{0, 1, 2, 3, 4, 5}

Tunable Parameters

→ Important Reminder! ←

Avoid Overfitting!

Tunable Parameters

maxBins:

- Number of bins used
- Transforming continuous features into discrete features
- ≥ max number of categories for any categorical feature

Tunable Parameters

Impurity:

- Measure used to choose between candidate splits
- Used in the information gain calculation
- Must match type of DecisionTree.

Lesson Objectives

- Investigate the different types of parameters involved in creating Decision Trees.
  - Specifiable Parameters (without Tuning required)
  - Tunable Parameters
  - Stopping Parameters (Tunable)
- Understand how varying the value used in parameters can affect the model positively or negatively

Decision Tree Parameters

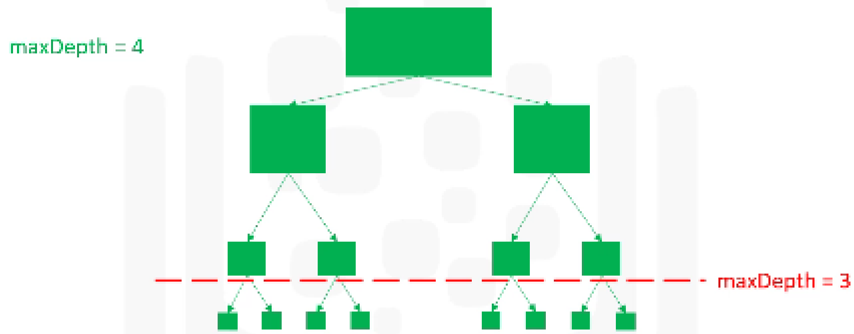
Specifiable Parameters (without Tuning required)
- numClasses, categoricalFeaturesInfo

Tunable Parameters
- maxBins, impurity

Stopping Parameters (Tunable)
- maxDepth, minInstancesPerNode, minInfoGain

Stopping Parameters (Tunable Parameters)

maxDepth = Maximum depth of the tree

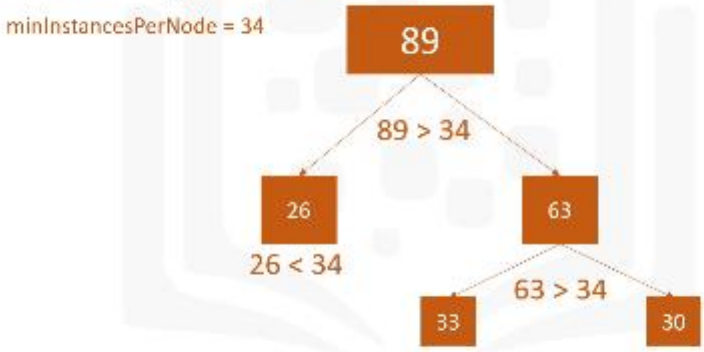


Larger Depth Yields:
- Chance of higher accuracy
- Increased Training Cost
- Risk of overfitting

Stopping Parameters (Tunable Parameters)

minInstancesPerNode:

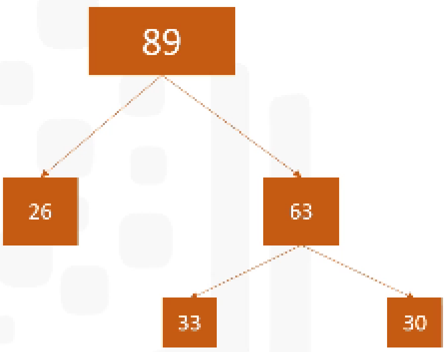
- Number of instances required for node to split further
- Commonly used in RandomForest



Stopping Parameters (Tunable Parameters)

minInfoGain:

- Node must provide at least this much information gain in order to split
- Information Gain: Amount of variability decrease resulting from a node split.



Lesson Objectives

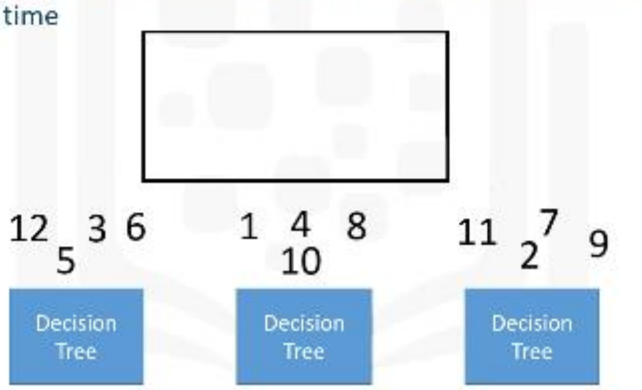
- Investigate the different types of parameters involved in creating Decision Trees and Random Forests.
  - Specifiable Parameters (without Tuning required)
  - Tunable Parameters
  - Stopping Parameters (Tunable)
- Examine how varying the value used in parameters can affect the model positively or negatively
- Describe the similarities in parameters between Decision Trees and Random Forests

Parameter Comparison

	Decision Tree Parameters	Random Forest Parameters
Specifiable Parameters (without Tuning required)	•numClasses •categoricalFeaturesInfo	•numClasses •categoricalFeaturesInfo •seed
Tunable Parameters	•maxBins •Impurity	•maxBins •impurity •numTrees •featureSubsetStrategy
Stopping Parameters (Tunable)	•maxDepth •minInstancesPerNode •minInfoGain	•maxDepth

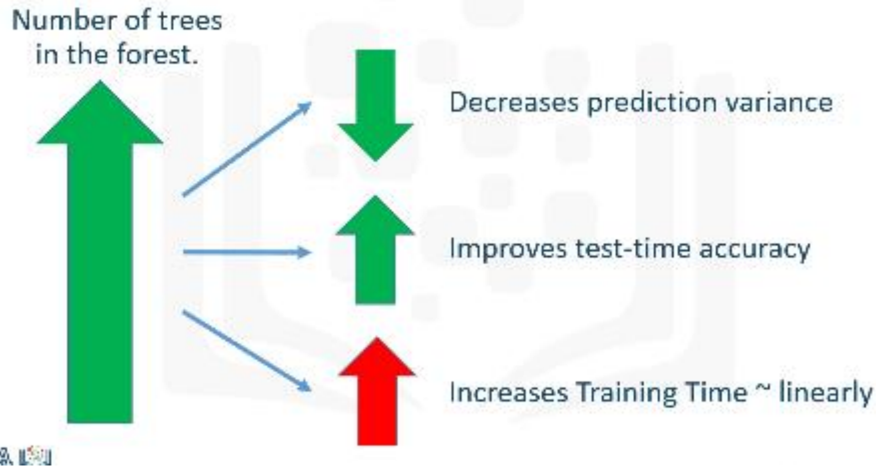
Specifiable Parameter (without Tuning)

seed:
- Random seed for bootstrapping, choosing feature subsets.
- Set as None (default): Generates seed based on system time



Tunable Parameters

numTrees: Defines Number of Trees in Forest



Tunable Parameters

featureSubsetStrategy:

- Number of features used as candidates for splitting at each tree node
- Specified as a function of the total number of features
- Decreasing speeds up training, but too low can affect performance
- Supports: "auto", "all", "sqrt", "log2", "onethird"

auto: Choose automatically for task:
If numTrees == 1, set to "all."
If numTrees > 1 (forest), set to "sqrt" for classification
If numTrees > 1 (forest), set to "onethird" for regression."

all: use all features

sqrt: use $\sqrt{\# \ of \ features}$

log2: use $log_2(\# \ of \ features)$

onethird: $\frac{\# \ of \ features}{3}$

Stopping Parameters (Tunable Parameters)

maxDepth: Maximum depth of each tree in the forest

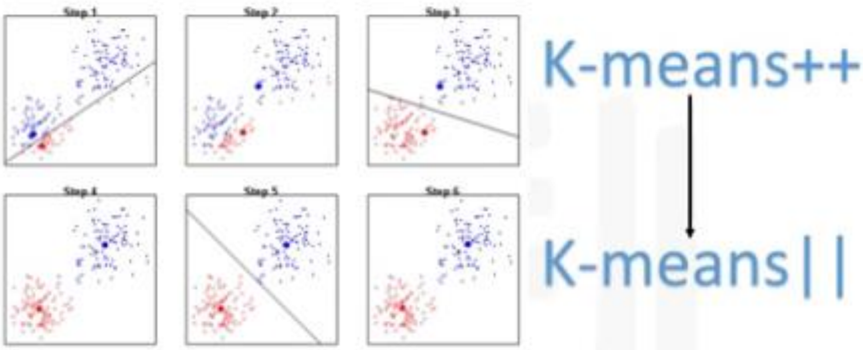
- Increasing maxDepth = more powerful and expressive model
- May increase training time
- May become more prone to overfitting.
- Averaging multiple trees yields variance reduction, allowing deeper trees

Lesson Objectives

K-Means Clustering

- Parallelized version of K-means through Spark MLlib
- Input parameters in K-means:
  - How these parameters affect the output
  - MaxIterations
  - Epsilon
- Available functions for use with K-means

K-Means Clustering



K-Means|| Parameters

classmethod train(*rdd, k, maxIterations=100, runs=1, initializationMode='k-means||', initializationSteps=5, epsilon=0.0001, initialModel=None*)

k: number of desired clusters

maxIterations: maximum number of iterations to run

runs: number of times to run, yielding best result out of the runs

initializationMode: specifies either random initialization or k-means|| initialization

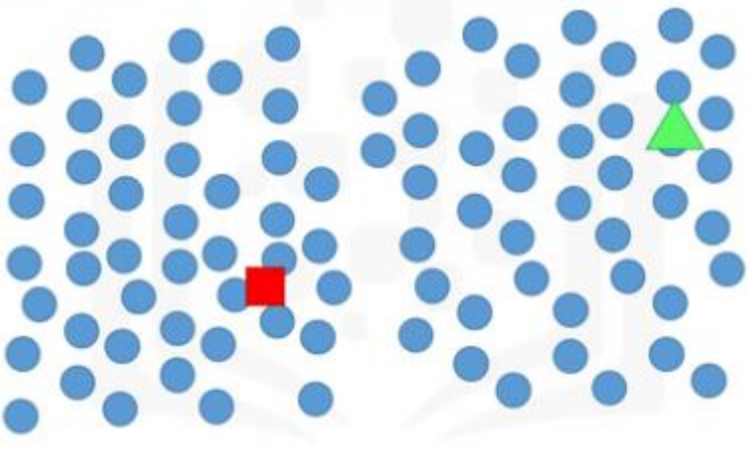
initializationSteps: determines number of steps in k-means|| algorithm

epsilon: determines the distance threshold within which we consider k-means to have converged

initalModel: optional set of cluster centers used for initialization. If set to true, only one run is performed

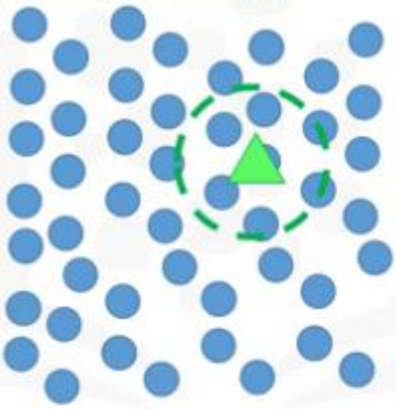
K-Means|| - maxIterations

maxIterations: maximum number of iterations to run



K-Means|| - epsilon

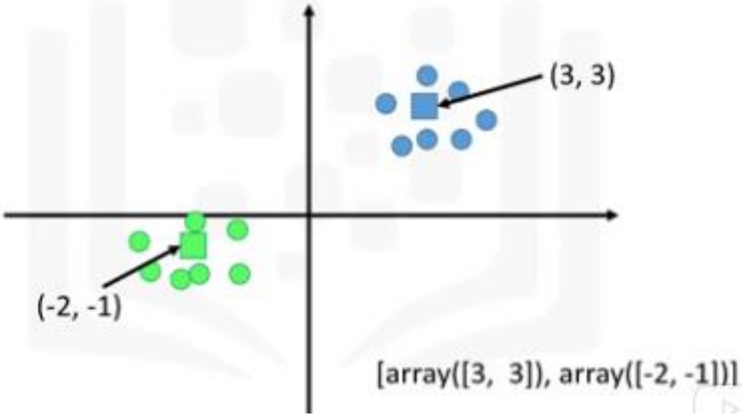
epsilon: determines the distance threshold within which we consider k-means to have converged



K-Means|| - Functions

clusterCenters

Returns cluster centers, represented as NumPy arrays.



K-Means|| - Functions

computeCost(rdd)

Returns the K-means cost (sum of squared distances of points to their nearest center) for this model on the given data.

k

The total number of clusters.

load(sc, path)

Load the model to a given path.

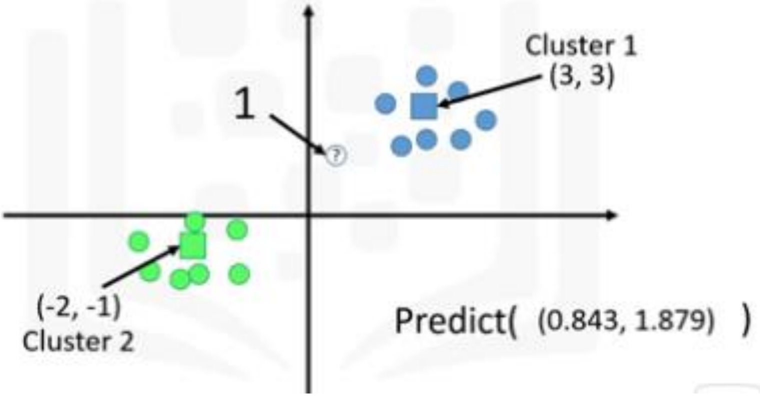
save(sc, path)

Save the model to a given path.

K-Means|| - Functions

predict(x)

Find the cluster that each of the points belongs to in this model.

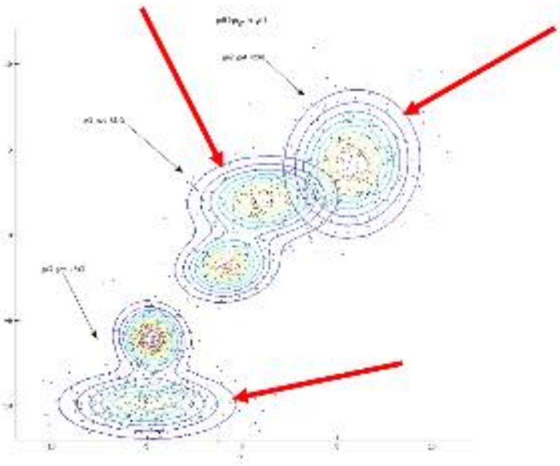


Lesson Objectives

Gaussian Mixture Clustering Algorithm:

- Parameters involved in creating a Gaussian Mixture
- Functions associated with Gaussian Mixture

Gaussian Mixture Clustering



Gaussian Mixture - Parameters

classmethod train(*rdd*, *k*, *convergenceTol=0.001*, *maxIterations=100*, *initialModel=None*)

k: number of desired clusters

convergenceTol: max change in log-likelihood where convergence is achieved

maxIterations: max number of iterations to perform without reaching convergence

initialModel: optional starting point for GMM algorithm. If omitted, a random starting point is constructed from the data.

Gaussian Mixture - Functions

gaussians

- Returns Array of MultivariateGaussian
- gaussians[i] - Multivariate Gaussian Distribution for the "i" Gaussian

Mean

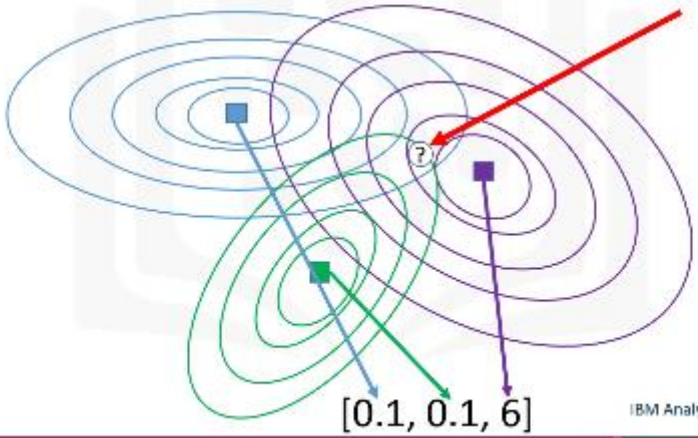
```
MultivariateGaussian(mu=DenseVector([-4.0109, -2.0483]),
                     sigma=DenseMatrix(2, 2, [1.222, -0.0085, -0.0085, 1.2463], 0))
```

Standard Deviation

Gaussian Mixture- Functions

predictSoft(x)

Returns the membership of point 'x' or each point in RDD 'x
to all mixture components



Gaussian Mixture - Functions

weights

- Weights for each Gaussian distribution in the mixture
- weights[i] - The weight for Gaussian "i"
- weights.sum == 1.

[0.1543518, 0.8456482]

Gaussian Mixture - Functions

k

Number of gaussians in mixture.

load(sc, path)

Load the model to a given path.

save(sc, path)

Save the model to a given path.