Big Data Analytics and Reasoning - Practice 07

Giuseppe Mazzotta

Machine Learning

Machine learning is a process for extracting patterns from your data, using statistics, linear algebra, and numerical optimization

Supervised Machine Learning vs Unsupervised Machine Learning

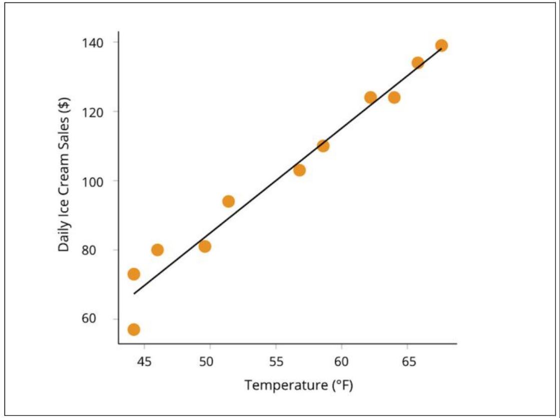
Spark MIlib - Applying machine learning algorithm, model evaluation, hyperparameters tuning ...

1. Intro

- **Supervised Learning**
  Input: set of labeled records
  Challenge: learning a model able to label unlabeled input
  - **Classification**
    Label is categorical - Binary or Multiclass Classification
  - **Regression**
    Predict a continuous value for a given record
- **Unsupervised Learning**
  Input: set of records
  Challenge: clustering records that share common patterns

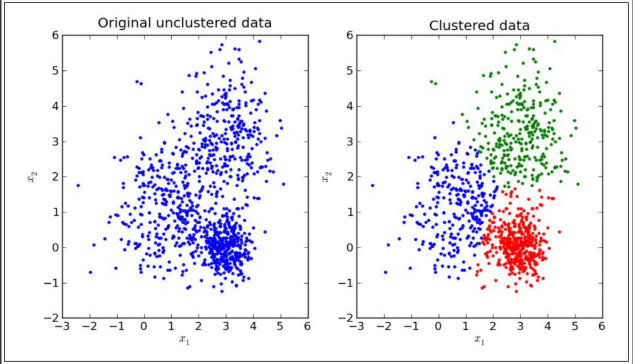
Examples!


Multinomial classification example: Australian shepherd, golden retriever, or poodle



Regression example: predicting ice cream sales based on temperature


Binary classification example: dog or not dog



Clustering example

2. Key Concepts

org.apache.spark.ml **package**

➔ **Transformer**

Transforms a dataset into a new one with one or more extra columns by applying rule-based transformation.
It has a *transform()* method

➔ **Estimator**

Learns parameters from a dataset and return a *Model* that is a Transformer. It has the *fit()* method

➔ **Pipeline**

It's an Estimator made by a series of Transformers and Estimators

Data Preparation

Most of the ML algorithms provided by Spark MLlib require two columns:

- *label* (Supervised case)
  Contains the value that we want to predict
- *features*
  It's a list representation of the features columns for a precise record

VectorAssembler merge different columns into a new one that store a vector of values

```
Source: plain csv file with 4 columns

    _c0, _c1, _c2, _c3
      0,   3,   1,   3
      1,   1,   1,   2
      0,   4,   1,   1
      0,   5,   1,   3
      1,   2,   1,   2

Predict the value of _c0 considering columns _c1 and _c3

    Transformation

        _c0, _c1, _c2, _c3, features, label
          0,   3,   1,   3,    [3,3],     0
          1,   1,   1,   2,    [1,2],     1
          0,   4,   1,   1,    [4,1],     0
          0,   5,   1,   3,    [5,3],     0
          1,   2,   1,   2,    [2,2],     1
```

Tip

Categorical features must be transformed into numerical without introducing dependency

Data Preparation

How to deal with categorical features?

- String column -> Integer column
  Enumerating possible strings from 1 to #possibleValues.
  WARNING we are introducing dependency
- Integer column -> Vector column
  One hot encoding technique. Each value is mapped to a list of size #possibleValues. This list contains 1 at value index and all the other 0
  WARNING possible waste of space

Spark uses ***SparseVector*** against ***DenseVector***

```
Feature to transform:
    category
        possible values motorcycle, car, truck

    First step
        motorcyle -> 1
        car -> 2
        truck -> 3

    Second step
        motorcyle -> [1,0,0]
        car -> [0,1,0]
        truck -> [0,0,1]

Source:
    vehicle_id, category, sits, price
    1,motorcycle,2,4
    2,car,5,20
    3,motorcycle,2,8
    4,truck,3,80

Transformation
    DenseVector representation

    vehicle_id, category, sits, price, label, features
    1,motorcycle,2,4 ,4 ,[1,0,0,2]
    2,car       ,5,20,20,[0,1,0,5]
    3,motorcycle,2,8 ,8 ,[1,0,0,2]
    4,truck     ,3,80,80,[0,0,1,3]

    SparseVector representation

    vehicle_id, category, sits, price, label, features
    1,motorcycle,2,4 ,4 ,[4,{1,4},{1,2}]
    2,car       ,5,20,20,[4,{2,4},{2,5}]
    3,motorcycle,2,8 ,8 ,[4,{1,4},{1,2}]
    4,truck     ,3,80,80,[4,{3,4},{1,3}]
```

Step 1

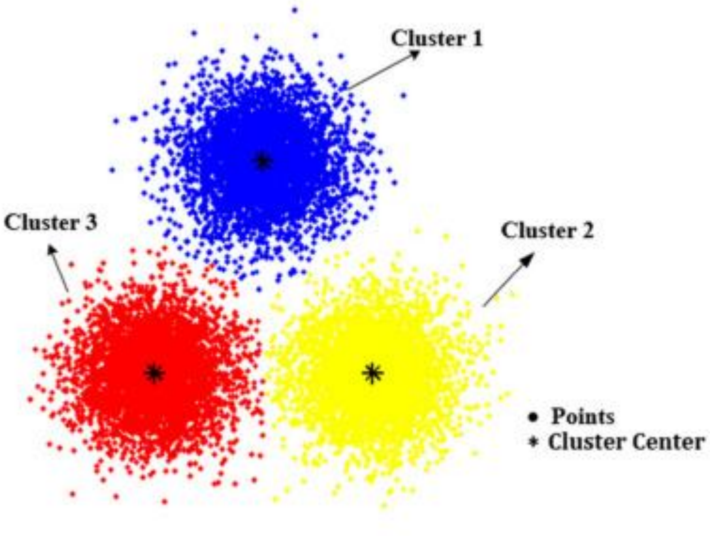
Step 2

Standard representation

In questo modo risparmiamo spazio

KMeans

K non si traina



Unsupervised clustering algorithm

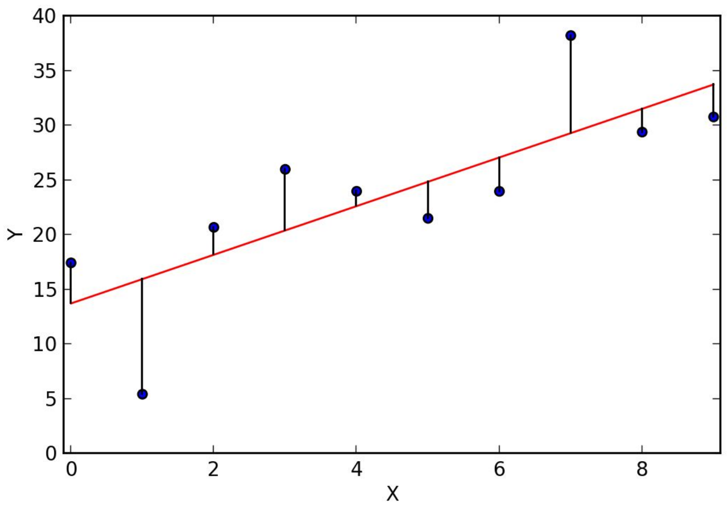
Objective: group data into cluster in such a way that similarly instances are in the same cluster

Quality evaluation

Silhouette measure that ranges from -1 to 1. For value near to 1 it means that instances in a cluster are really close to each other and are also really far from instance inside other clusters

Note is not easy to estimate the correct number of clusters, the result strongly depend to the centroid initialization

Linear Regression



Used to predict continuous values

Objective: finding a linear relation between label and features that approximates all the points into training set

Quality evaluation

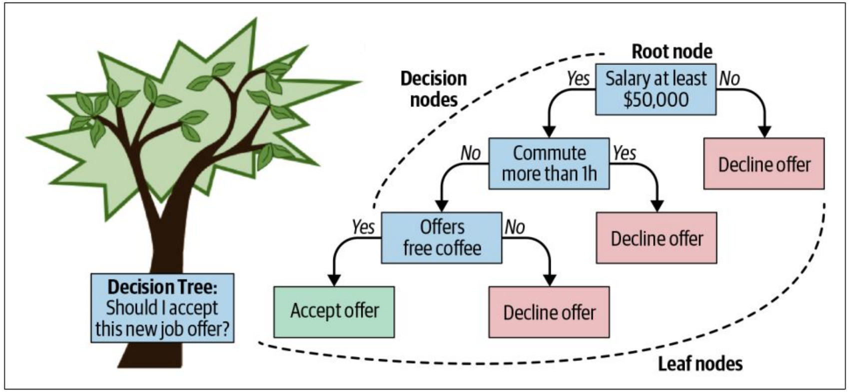
RMSE: Root Mean Squared Error (RMSE) $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$

R2: $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ where $SS_{tot} = \sum_{i=1}^{n}(y_i - \bar{y})^2$

$SS_{res} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Più R^2 è vicino a 1, meglio è

Decision Tree



Used both in classification and regression

Objective: learn patterns among data to predict class/numeric value

Decision trees are trees where:

- Each node represents a condition on one column value
- A leaf node represents the prediction for a record that matches the conditions on the path that reach the precise leaf node

Quality evaluation

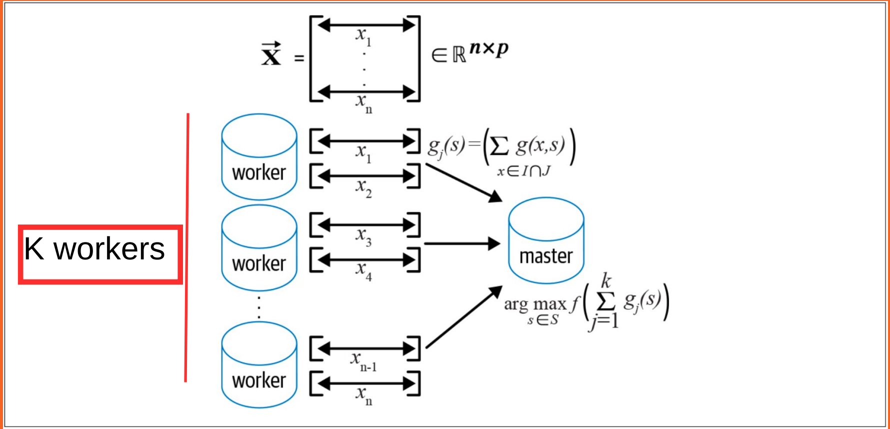
RMSE, **R2** for Regression problem
Accuracy percentage of correctly classified instances for classification problem and more …

Distributed Training for DT

Training data is partitioned among workers (more detail [here](#))

How to determine splitting condition at i-th node?

- *Worker computes static on its own partition w.r.t. current dt's node and possible split conditions*
- *Worker statistics are sent to master (driver program) that computes the optimal split condition*
- The optimal split is sent to workers to update their internal state



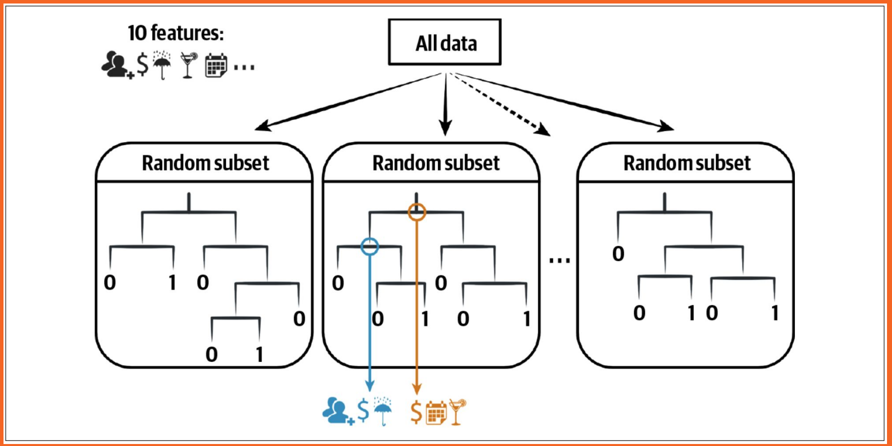
Tip

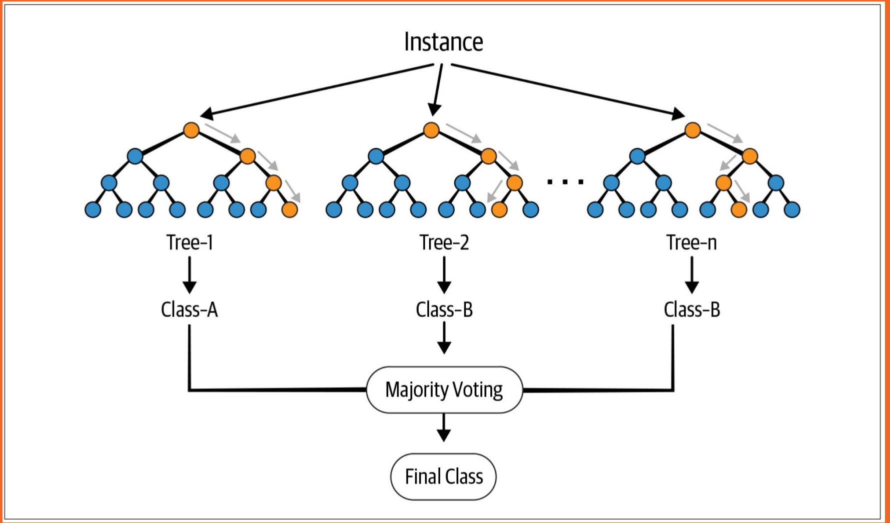
Warning: *maxBins* determines the number of bins to discretize continuous values. Check it is sufficient also for discrete columns

Random Forest

It is an ensemble of decision trees

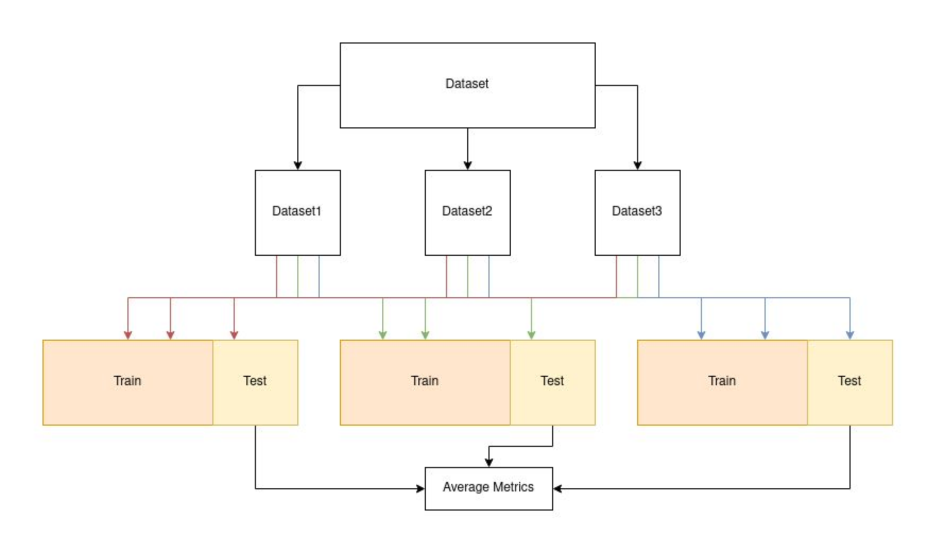
- Prediction depends on the combination of prediction of each decision trees
- Problem: each decision tree is likely to learn same patterns in the data:
  - Bootstrapping sample by rows
  - Random feature selection by columns
- Learn different “*weak*” trees to build a more robust ensemble
- How to estimate the number of decision trees or the maximum depth for each tree





Hyperparameter Tuning

- K-Fold Cross Validation
  Training data is split in k folds
  For *i* in {1, ..., K}:
  Train model on fold_j, j!=i
  Test model on fold_i
  Compute average metrics
- For each possible hyperparameter value executes k-fold cross validation
- The best metrics average determines the best hyperparameter value
- Train the model with the best hyperparameter value on the entire dataset



Let's practice with MLlib ....