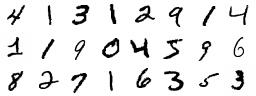
Saving & loading Machine Learning (ML) models

Our datasets are vectors of pixels representing images of handwritten digits. For example:



These datasets are stored in the popular LibSVM dataset format. We will load them using the Spark SQL data source for LibSVM files.

# Load Data:

**# We explicitly set numFeatures = 784 since the images are 28x28 = 784 pixels (features).**

**training = sqlContext.read.format("libsvm").option("numFeatures", "784").load("/databricks-datasets/mnist-digits/data-001/mnist-digits-train.txt")**

**# Cache data for multiple uses.**

**training.cache()**

**print "We have %d training images." % training.count()**

Show a sample of our data. Each image has the true label (the label column) and a vector of features which represent pixel intensities. Note the vectors are stored in sparse format.

**display(training)**

# Train a Random Forest Classifier

We train a random forest for classifying images of digits. Given a new image of a digit, this classifier will be able to predict the actual digit 0 - 9 in the image.

**from pyspark.ml.classification import RandomForestClassifier**

**# Run the Random Forest algorithm on our data**

**rf = RandomForestClassifier(numTrees=20)**

**model = rf.fit(training)**

# Save the Random Forest model

**basePath = "/tmp/mllib-persistence-example"**

**model.save(basePath + "/model")**

**# You may also specify "overwrite" just as when saving Datasets and DataFrames:**

**# model.write().overwrite().save(basePath + "/model")**

Scala: The data engineer

#### Set up

#### First create a temp folder which will be used for saving and loading models.

**val** basePath = "/tmp/mllib-persistence-example"

dbutils.fs.rm(basePath, recurse=true)

dbutils.fs.mkdirs(basePath)

# A data engineer now loads the model from the data scientist into a Scala workflow and tests it on new data.

**import org.apache.spark.ml.classification.RandomForestClassificationModel**

**val model = RandomForestClassificationModel.load(basePath + "/model")**

**val test = sqlContext.read.format("libsvm").option("numFeatures", "784").load("/databricks-datasets/mnist-digits/data-001/mnist-digits-test.txt")**

**val predictions = model.transform(test)**

**display(predictions.select("label", "prediction"))**