SolrClassification

Lucene Document Classification Integration

⚠ Solr6.1

Introduction

This documentation is about the integration of the Lucene Classification module with Solr to allow Solr users to easily manage simple Classification problems out of the box .

Solr Classification

The classification in Solr can happen 2 sides :
Indexing time - through an Update Request Processor Query Time - through a Request handler (similar to the More like This) This documentation is about Indexing time integration :
The Classification Update Request Processor.

Classification Update Request Processor

First of all let's describe some basic concepts :
An Update Request Processor Chain, associated to an Update handler, is a pipeline of Update processors, that will be executed in sequence. It takes in input the added Document (to be indexed) and return the document after it has been processed by all the processors in the chain in sequence. Finally the document is indexed. An Update Request Processor is the unit of processing of a chain, it takes in input a Document and operates some processing before it is passed to the following processor in the chain if any.

The main reason for the Update processor is to add intermediate processing steps that can enrich, modify and possibly filter documents , before they are indexed. It is important because the processor has a view of the entire Document, so it can operate on all the fields the Document is composed.

Description

The Classification Update Request Processor is a simple processor that will automatically classify a document (the classification will be based on the latest index available) adding a new field containing the class, before the document is indexed. After an initial valuable index has been built with human assigned labels to the documents, thanks to this Update Request Processor will be possible to ingest documents with automatically assigned classes. The processing steps are quite simple :
When a document to be indexed enters the Update Processor Chain, and arrives to the Classification step, this sequence of operations will be executed :
The latest Index Reader is retrieved from the latest opened Searcher A Lucene Document Classifier is instantiated with the config parameters in the solrconfig.xml A Class is assigned by the classifier taking in consideration all the relevant fields from the input document A new field is added to the original Document, with the class The Document goes through the next processing step

Configuration

Let's see the detailed configuration for the Update Processor with examples :

e.g.

K Nearest Neighbours Classifier

```
<updateRequestProcessorChain name="classification">
<processor class="solr.ClassificationUpdateProcessorFactory">
<str name="inputFields">title^1.5,content,author</str>
<str name="classField">cat</str>
<str name="algorithm">knn</str>
<str name="knn.k">20</str>
<str name="knn.minTf">1</str>
<str name="knn.minDf">5</str>
</processor>
</updateRequestProcessorChain>
```

e.g.

Simple Naive Bayes Classifier

```
<updateRequestProcessorChain name="classification">
<processor class="solr.ClassificationUpdateProcessorFactory">
<str name="inputFields">title^1.5,content,author</str>
<str name="classField">cat</str>
<str name="algorithm">bayes</str>
</processor>
</updateRequestProcessorChain>
```

e.g.

Update Handler Configuration

```
<requestHandler name="/update" >
<lst name="defaults">
<str name="update.chain">classification</str>
</lst>
</requestHandler>
```

Parameter	**Default**	**Description**
inputFields	This config param is mandatory	The list of fields (comma separated) to be taken in consideration for doing the classification. Boosting syntax is supported for each field.
classField	This config param is mandatory	The field that contains the class of the document. It must appear in the indexed documents. If knn algorithm it must be stored. If bayes algorithm it must be indexed and ideally not heavily analysed.
predictedClassField	classField	The field that will store the ouput of the classification (the predicted class)
predicatedClass.maxCount	1	The max number of classes to assign. It will represent the Top K among the candidates
algorithm	knn	The algorithm to use for the classification: - knn (K Nearest neighbours) - bayes (Simple Naive Bayes)
knn.k	10	Advanced - the no. of top docs to select in the MLT results to find the nearest neighbor
knn.minDf	1	Advanced - A term (from the input text) will be taken in consideration by the algorithm only if it appears at least in this minimum number of docs in the index
knn.minTf	1	Advanced - A term (from the input text) will be taken in consideration by the algorithm only if it appears at least this minimum number of times in the input

Usage

Indexing News Documents ? we can use the already indexed news with category, to automatically tag upcoming stories with no human intervention. E-commerce Search System ? Category assignation will require few human interaction after a valid initial corpus of products has been indexed with manually assigned category. The possible usage for this Update Request Processor are countless. In any scenario where we have documents with a class or category manually assigned in our Search System, the automatic Classification can be a perfect fit. Leveraging the existent Index , the overhead for the Classification processing will be minimal. After an initial human effort to have a good corpus of classified Documents, the Search System will be able to automatically index the class for the upcoming Documents. Of course we must remember that for advanced classification scenarios that require in deep tuning, this solution can be not optimal.

[1] http://www.slideshare.net/AlessandroBenedetti/lucene-and-solr-document-classification
[2] http://alexbenedetti.blogspot.co.uk/2015/07/lucene-document-classification.html
[2] http://www.slideshare.net/teofili/text-categorization-with-lucene-and-solr