Information Retrieval - Assignment 2

Main program

assignment2.Main requires following arguments:

-trainData [directory]	(directory for trainData)
-testData [directory]	(directory for testData)
-labeled [true|false]	(does testData contains labels/topics, if yes then true otherwise false)
-type [NB|LR|SVM]	(NB for NaiveBayse, LR for Logistic Regression, SVM for Support Vector Machines)

For instance:

-trainData C:/IR/trainData/ -testData C:/IR/test-with-labels/ -labeled true -type SVM

It's important to set following VM Arguments:

-Xss400m -Xms2g -Xmx4g -XX:-UseGCOverheadLimit

General Classification Information

All 3 classification are using one-vs-all approach.

All 3 classification are using StopWords (assignment2.StopWords.scala) and Stemming (com.github.aztek.porterstemmer.PortStemmer.scala)

For all 3 classification top 3 topics are returned.

Naive Bayse

Class: assignment2.naivebayse.NaiveBayseClassification

In a first pass a assignment2.index.IndexBuilder collects all relevant information from train data, such as nr of documents, topic counts, topic length (total number of tokens for each topic) and topicTfIndex (collection frequency for each topic), and puts it in Memory.

In a second pass NaiveBayseClassification goes over test data

Formula Slide

Cyrill Zadra

Best result using Naive Bayse:

P= 0.7194131709337228 , R= 0.7333289634183215 , F1= 0.7020213093418058

Logistic Regression

Class: assignment2.regression.LogisticRegressionClassification

In a first pass LogisticRegressionClassification uses assignment2.index.FeatureBuilder to collect separately all features (term frequencies) from train and test data.

In training step:

For each topic (theta) in train data SVM goes over a number (NUMBER_OF_ITERATIONS) of randomly picked train features and updates vector theta.

Best result using Logistic Regression:

P= 0.13322248487513225 , R= 0.18784543859165262 , F1= 0.1502050493620021

SVM - Support Vector Machines

Class: assignment2.svm.SvmClassification

In a first pass SvmClassification uses assignment2.index.FeatureBuilder to collect separately all features (term frequencies) from train and test data.

In training step:

For each topic (theta) in train data SVM goes over a number (NUMBER_OF_ITERATIONS) of randomly picked train features and updates vector theta.

$$\vec{\theta}_{t+1} = \begin{cases} (1-\eta_t\lambda)\vec{\theta}_t & \text{if } y_{I(t)}\langle\vec{\theta}, \vec{x}_{I(t)}\rangle \geq 1 \\ (1-\eta_t\lambda)\vec{\theta}_t + \eta_t y_{I(t)}\vec{x}_{I(t)} & \text{otherwise} \end{cases}$$

In prediction step:

For each test document SVM goes over all topic thetas and computes hingeLoss. Top 3 scores are returned.

$$\max\{0, 1 - y\langle\vec{\theta}, \vec{x}\rangle\}$$

Best result using SVM:

P= 0.5904149471800447 , R= 0.601059109400312 , F1= 0.5744665782352627