Paper SAS634-2017

An Efficient Way to Deploy and Run Text Analytics Models in Hadoop

Seung Lee, Xu Yang, and Saratendu Sethi, SAS Institute Inc.

ABSTRACT

Significant growth of the Internet has created an enormous volume of unstructured text data. In recent years, the amount of this type of data that is available for analysis has exploded. While the amount of textual data is increasing rapidly, an ability to obtain key pieces of information from such data in a fast, flexible, and efficient way is still posing challenges. This paper introduces SAS® Contextual Analysis In-Database Scoring for Hadoop, which integrates SAS® Contextual Analysis with the SAS® Embedded Process. SAS® Contextual Analysis enables users to customize their text analytics models in order to realize the value of their text-based data. The SAS® Embedded Process enables users to take advantage of SAS® In-Database Code Accelerator for Hadoop to run scoring models. By using these key SAS® technologies, the overall experience of analyzing unstructured text data can be greatly improved. The paper also provides guidelines and examples on how to deploy and run category, concept, and sentiment models for text analytics in Hadoop.

INTRODUCTION

With the rapid development of digital information technologies, enterprises are able to collect and store large amounts of data. The data size could be as large as terabytes and petabytes. Among the collected data, unstructured (primarily text) data accounts for more than 80 percent and is growing at an exponential rate. The unstructured content can be found from different sources, for example, blogs, forums, daily logs, product reviews, call center logs, emails, customer reviews and other forms of electronic text. Opportunities for analyzing such textual data to reveal insightful information for improving business operations and performance are attractive. Text analytics is the mechanism that enables organizations to automate the identification of topics, entities, facts, and events, coupled with sentiment and other subjective information.

In order to analyze the volume of data without consuming an excessive amount of network resources, IT professionals seek tools that empower them to deploy analytical algorithms inside of their Hadoop infrastructure.

In this paper, we presents an efficient and seamless approach to deploy and score SAS Text Analytics models in Hadoop. By using key SAS technologies, large amounts of unstructured text data are analyzed and harnessed so that you can understand the value of the data. SAS Contextual Analysis lets the user customize and build text analytics scoring models. These models can then be published to a Hadoop cluster. SAS DS2 is the programming language used to execute the text analytics models in Hadoop from the client machine. SAS In-Database Code Accelerator for Hadoop and SAS Embedded Process enable the DS2 program to run in Hadoop.

This paper is structured as follows:

- The basic components and framework of SAS Contextual Analysis In-Database Scoring for Hadoop technology are discussed.
- A concrete example to show how you can use the SAS DS2 program and text analytics models to score text data is provided.

OVERVIEW OF SAS® CONTEXTUAL ANALYSIS IN-DATABASE SCORING FOR HADOOP

As shown in Figure 1, SAS Contextual Analysis generates binary text analytics models and DS2 score code. The saved binary models are deployed to Hadoop cluster data nodes to analyze data in Hadoop. The extracted DS2 code requires modifications in order to satisfy the configuration of specific Hadoop distribution versions and the location of deployed text analytics models. Details of DS2 modifications are explained in Section: Running SAS Text Analytics in Hadoop.

The SAS In-Database Code Accelerator for Hadoop enables you to publish a DS2 threaded program and its associated files to the database, and then execute the program in parallel within SAS Embedded Process. When the DS2 code is executed, a MapReduce job is created and run using the YARN resource manager. The MapReduce job uses the SAS Text Analytics components in SAS Embedded Process, which runs where the data resides.

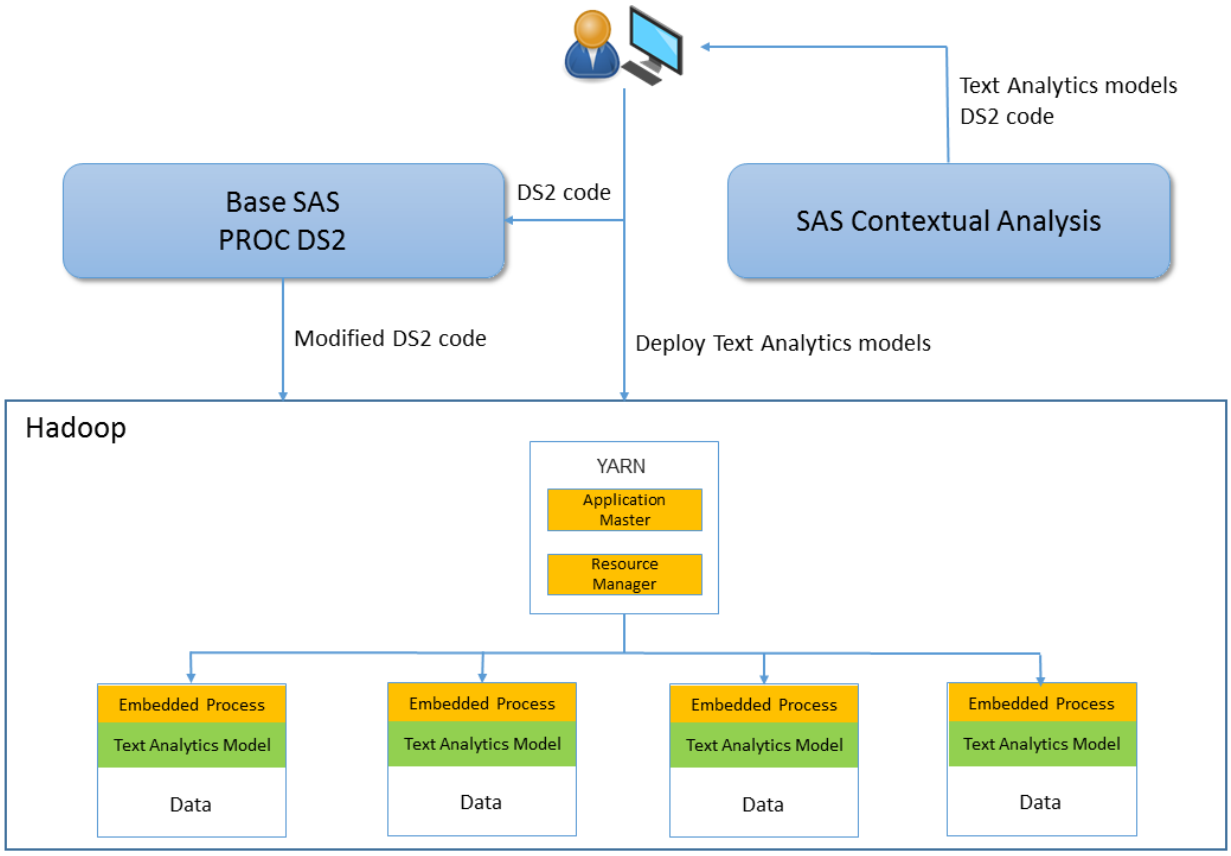


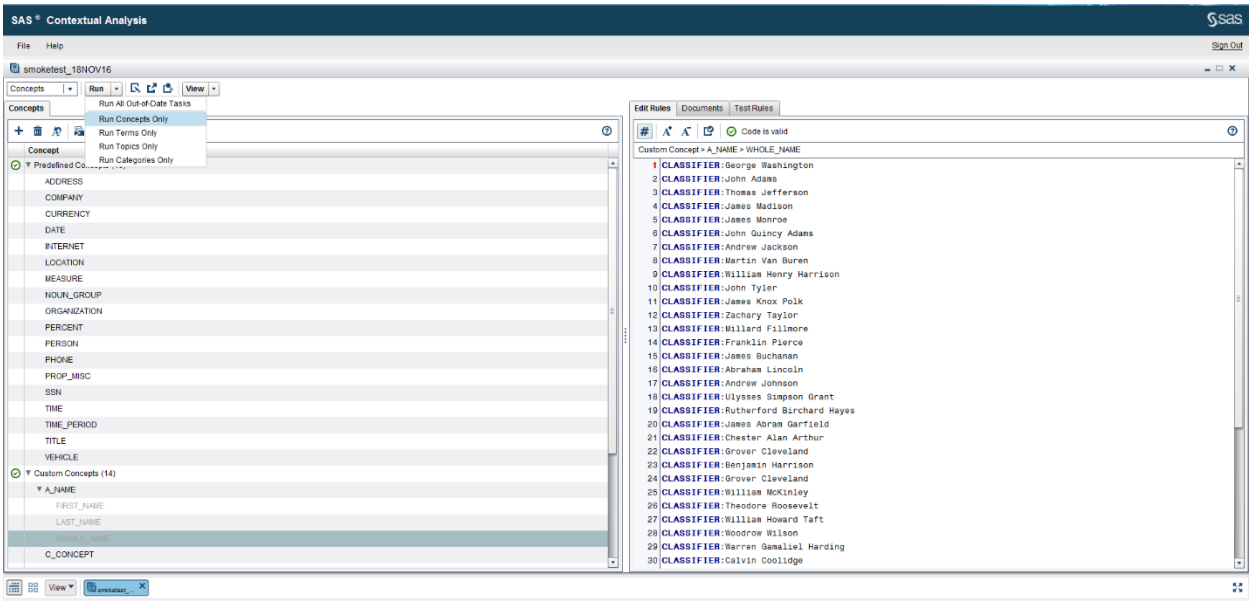
Figure 1. Framework of SAS Contextual Analysis In-Database Scoring for Hadoop

SAS CONTEXTUAL ANALYSIS

SAS Contextual Analysis is a web-based text analytics application that uses natural language processing and machine learning to derive insights from textual data. Using this application, you can determine topic identification, categorization, entity and fact extraction, and sentiment analysis in a single user interface. The application enables you to build models (based on training documents) and create taxonomies and rule sets to analyze documents. You can then customize your models for your business domain in order to realize the value of your text-based data (Bultman 2016).

At the end of the modeling process, SAS Contextual Analysis generates DS2 code and binary text analytics models for scoring text data. SAS Contextual Analysis generates three types of DS2 score code and models corresponding to categorization, concept extraction, and sentiment analysis. The DS2 code can be run within a SAS environment such as SAS® Studio or within Hadoop using SAS In-Database Code Accelerator for Hadoop. The binary models represent the rule sets for categorization (file extension: .mco), concepts (file extension: .li) and sentiment (file extension: .sam) taxonomies, which are highly optimized to apply all rules in parallel.

Display 1 shows the concept model build options in SAS Contextual Analysis.



Display 1. Building a Concept Model in SAS Contextual Analysis

DEPLOYING SAS TEXT ANALYTICS SCORING MODELS

To deploy a SAS Text Analytics scoring model generated from SAS Contextual Analysis, specific steps must be followed, as shown below. The overall process is illustrated in Figure 2.

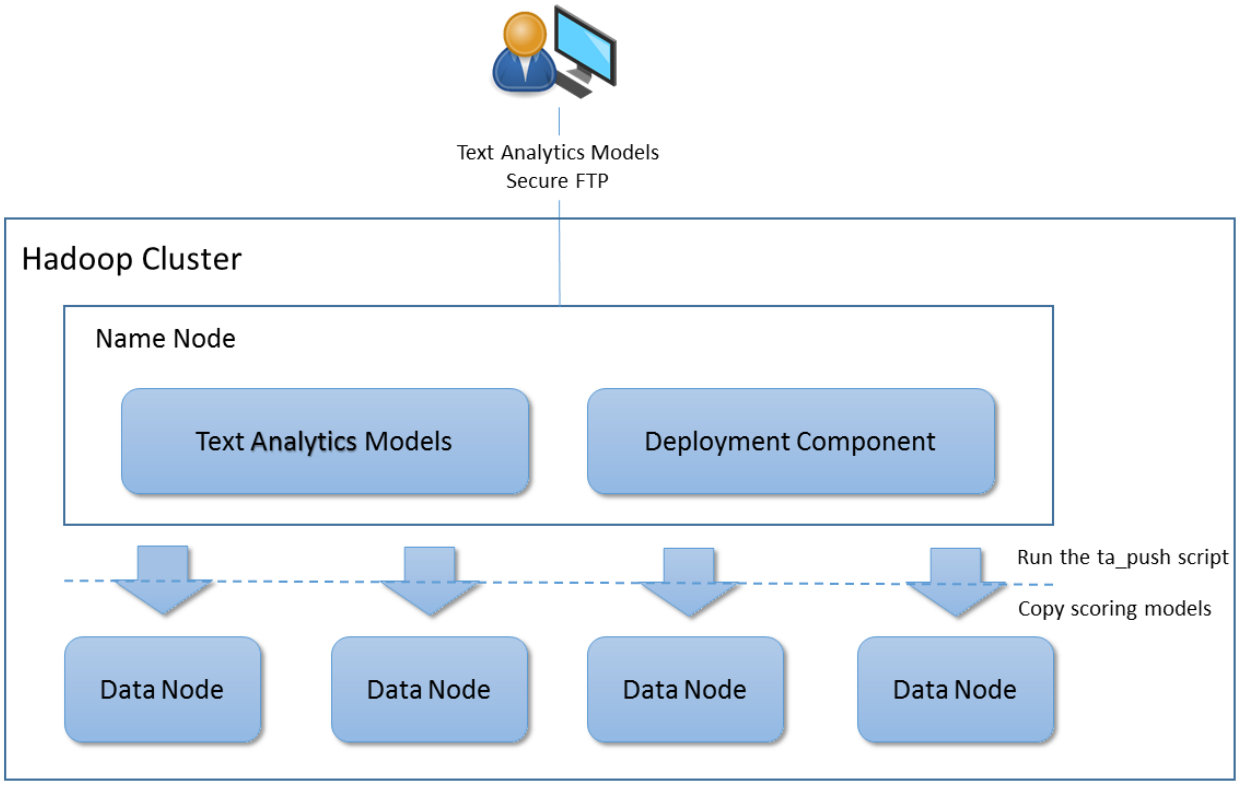


Figure 2. Deploying SAS Text Analytics Scoring Models

1. Obtain a SAS Text Analytics model for scoring concept extraction (.li file), categorization (.mco file),

or sentiment analysis (.sam file).

2. Copy the scoring code model to the Hadoop name node. After you have obtained a SAS Text Analytics model, you must copy it to the Hadoop name node. It is recommended that you copy the model to a temporary staging area, such as **`/tmp/tastage`**. You can copy the model to the Hadoop Name Node by using a file transfer command such as FTP or SCP.

3. Use the ta_push.sh script to deploy the SAS Text Analytics score code mode in the cluster. SAS Contextual Analysis In-Database Scoring for Hadoop provides a ta_push.sh executable file to enable you to deploy the SAS Text Analytics models on Hadoop cluster nodes. The ta_push.sh file copies the specified model to a user-specified location on each of the Hadoop nodes. The ta_push.sh file automatically discovers all nodes in the cluster and deploys the model to the specified target model path on each of the cluster data nodes.

   The ta_push.sh file must be run as the root user. The root user becomes the HDFS user in order to detect the nodes in the cluster. Here is an example of running the script:

```
cd EPInstallDir/SASEPHome/bin
./ta_push.sh –s /tmp/tastage/en-ne.li –t /opt/sas/ta/model/en-ne.li
```

SAS DS2 LANGUAGE

DS2 is a procedural programming language with variables and scope, methods, packages, control flow statements, table I/O statements, and parallel programming statements. A DS2 program can be run in various SAS products, including Base SAS®, SAS® High-Performance Analytics, SAS® In-Database Code Accelerator for Hadoop, and SAS® In-Memory Analytics. The core of DS2 is similar to SAS DATA step language. Expressions in DATA step operate the same way in DS2 and most DATA step functions can be called within DS2.

DS2 has programming blocks to enable you to perform specific tasks in a modular fashion. The programming blocks are automatically invoked as DS2 executes the program.

- **Variable** - Variables are declared at the top of a DS2 program and are considered global variables having global scope (within the program).
- **Package** - The threaded-kernel (TK) extension package variables are declared at the top of the DS2 program. The package variables have global scope (within the program).
- **Methods** - DS2 classifies initialization, processing, and termination phases using three system-defined methods: init(), run(), and term(), respectively.
  - The init() method is called at the start of the program. The method is used to initialize variables and invoke the package initialization process.
  - The run() method is called after the init() method. The method iterates on all observations in the input data set.
  - The term() method is called after the run() method completes its processes. The method performs any wrap-up processing and outstanding resource cleanup.
- **Thread** - Concurrent processing is allowed using a threaded program. The threaded programming block must be declared with the `thread` command and must end with `endthread`.
- **Data program** - The block of code declared with the DATA step statement.

The programming block defines the scope of the declared variables. A variable in a nested programming block has local scope while a variable in the outermost programming block has global scope.

SAS EMBEDDED PROCESS

SAS Embedded Process is a lightweight execution container that is easily deployable on a variety of platforms. SAS Embedded Process enables DS2 code to run inside Hadoop and effectively leverages the massive parallel processing of Hadoop. The process must be installed on every node of the Hadoop

cluster to capitalize on its parallelism while running DS2 code, where the process acts as a MapReduce application. Therefore, all computing resources used by SAS Embedded Process are manageable by the YARN resource manager (Ghazaleh 2016).

SAS Embedded Process includes SAS Text Analytics components where it consists of natural language processing thread-kernel (TK) extensions. The components enable you to execute text analytics scoring models in Hadoop. The same threaded-kernel extensions are installed in SAS Contextual Analysis.

RUNNING SAS TEXT ANALYTICS IN HADOOP

Understanding the DS2 code necessary to run SAS Text Analytics models inside Hadoop is simple. However, it involves understanding particular features that are related to Hadoop environment settings. This section presents the steps needed to properly configure Hadoop environment variables. It also presents a complete example of the process for writing DS2 code for scoring SAS Text Analytics.

SETTING SAS ENVIRONMENT VARIABLES

In order to run SAS Text Analytics models in Hadoop, the Hadoop Java client information must be specified in DS2 code. The Hadoop client Java Archive (JAR) files and XML client configuration files enable the client to connect the Hadoop services. Both JAR and XML files are specific to the version of the Hadoop distribution and provided by Hadoop vendors.

The Hadoop client JAR and services configuration files can be collected directly from the local file system in one of the Hadoop nodes. The configuration files are commonly stored under **`/etc/hadoop/conf`** (for Hadoop) and **`/etc/hive/conf`** (for Hive).

The Hadoop client files can also be collected using the SAS Deployment Manager. The SAS Deployment Manager requires you answer a set of questions that help locate Hadoop Java client files. For complete information about collecting Hadoop client files, see *SAS® Contextual Analysis In-Database Scoring for Hadoop: Administrator's Guide*. You must have system privileges to perform the deployment process.

After the Hadoop client JAR and services configuration files are obtained, you must set the following environment variables. The location that contains all of the Hadoop client JAR and services configuration files must be accessible from the client machine.

1. SAS_HADOOP_JAR_PATH: Specifies the location of Hadoop client JAR files. The location that contains Hadoop client JAR files must be accessible on the client machine. The variable is set by using an OPTION SET statement. The following shows sample code to set the Hadoop JAR location.

   ```
   options set=SAS_HADOOP_JAR_PATH="C:\your\Hadoop\jars\location";
   ```

2. SAS_HADOOP_CONFIG_PATH: Specifies the location of Hadoop services configuration files. The location that contains Hadoop services configuration files must be accessible on the client machine. The variable is set by using an OPTION SET statement. The following shows sample code to set the Hadoop services configuration location.

   ```
   options set=SAS_HADOOP_CONFIG_PATH="C:\your\Hadoop\config\location";
   ```

DS2 CODING PROCESS

The process of writing DS2 code for SAS Text Analytics models in Hadoop is outlined here.

1. Enable the SAS In-Database Code Accelerator to execute the DS2 threaded program inside Hadoop by using the following command:

   ```
   proc ds2 ds2accel=yes;
   ```

2. Set SAS environment variables that specify the location of the Hadoop Java client API and configuration files.

```
options set=SAS_HADOOP_JAR_PATH="C:\your\Hadoop\jars\location";

options set=SAS_HADOOP_CONFIG_PATH="C:\your\Hadoop\config\location";
```

3. Use a LIBNAME statement to gain access to a Hadoop cluster to read and write data to and from Hadoop. The LIBNAME statement is used to assign a library reference to associate with a Hadoop HDFS or Hive server. SAS in-database technologies use the SAS/ACCESS Interface to Hadoop. SAS/ACCESS Interface to Hadoop provides enterprise data access and integration between SAS and Hadoop and works similarly to other SAS engines.

```
libname gridlib hadoop server="myhivenode.com" user=myuserid
password=mypassword;
```

4. Indicate where to find the binary files that are installed during the deployment of SAS Text Analytics scoring models. Depending on what scoring models you are using, look for one of following.

```
/** Concept model scoring binary**/
%let liti_binary_path = '/path/to/liti/binary.li';

/** Category model scoring binary**/
%let mco_binary_path = '/path/to/mco/binary.mco';

/** Sentiment model scoring binary**/
%let sam_binary_path = '/path/to/sam/binary.sam';
```

5. Define a threaded program to allow parallel execution.

```
THREAD workerth / overwrite=YES;
```

CONCEPT SCORING EXAMPLE

The following DS2 code is a complete program for concept model scoring. The category and sentiment model scoring are similar. You can also find SAS Text Analytics models scoring examples in *SAS® Contextual Analysis In-Database Scoring for Hadoop: User's Guide*.

```
/*************************************************************************
 * Set SAS environment variables that specify the location of the Hadoop
 * Java client API and configuration files. They are used to access Hadoop
 * services. The Java client API is provided by the Hadoop vendor in the
 * form of JAR files.
 *
 * NOTE: The SAS Contextual Analysis In-Database Scoring for Hadoop
 * Administrator's Guide describes how to collect Hadoop Java client API
 * JARs and configuration files.
*************************************************************************/
options set=SAS_HADOOP_JAR_PATH="C:\path\to\Hadoop\jars";
options set=SAS_HADOOP_CONFIG_PATH="C:\path\to\Hadoop\conf";

/*************************************************************************
 * Execute a LIBNAME statement to assign a library reference to associate
 * with a Hadoop HDFS or HIVE server.
 *
 * Please contact your IT administrator for the HIVE server name.
 *************************************************************************/
libname gridlib hadoop server="hivenode.com" user=userid password=password;

/*************************************************************************
 * Execute a LIBNAME statement to assign a library reference to the
 * location of the local SAS data set to be scored.
 *
 * If all the data sets used for scoring are already in HDFS, the library
 * might not be needed.
```

```
 ****************************************************************************/
libname home "C:\path\to\local\dataset";

/****************************************************************************
 * Copy the data set to HDFS. Unique observation IDs are required and are
 * created as the field _document_id.
 *
 * If all the data sets used for scoring are already in HDFS, this step
 * might not be needed.
 ****************************************************************************/
data gridlib.input_dataset;
  set home.input_dataset;
  _document_id = _N_;
Run

/****************************************************************************
 * Set input/output macro variables.
 *
 * input_ds: Name of the input data set in Hadoop, including the HDFS
libref
 * document_id: Name of the unique document ID column in the input data set
 * document_column: Name of the column to process in the input data set
 * liti_binary_path: Path to the LITI binary
 * output_ds: Name of the output data set, including the HDFS libref
 ****************************************************************************/
%let input_ds = gridlib.input_dataset;
%let document_id = _document_id;
%let document_column = text_to_process;
%let liti_binary_path = '/path/to/liti/binary.li';
%let output_ds = gridlib.concept_out;

/****************************************************************************
 * Delete the output before starting (Optional)
 ****************************************************************************/
proc delete data=&output_ds; run;

/****************************************************************************
 * Scores concepts in Hadoop
 ****************************************************************************/
proc ds2 ds2accel=yes xcode=warning;

  /* These packages are part of the Text Analytics add-on and are      */
  /* installed in the EP                                               */
  require package tkcat; run;
  require package tktxtanio; run;

  /* The output of the thread program is the input of the data program */
  THREAD workerth / overwrite=YES;

    dcl package tkcat cat();
    dcl package tktxtanio txtanio();
    dcl binary(8) _apply_settings;
    dcl binary(8) _document;
    dcl binary(8) _liti_binary;
    dcl binary(8) _trans;
    dcl double _status;
    dcl double _num_matches;
```

```
dcl double _i;
dcl double _document_id;
dcl varchar(1024) _name;
dcl varchar(1024) _full_path;
dcl double _start_offset;
dcl double _end_offset;
dcl varchar(1024) _term;
dcl varchar(1024) _canonical_form;
retain _apply_settings;
retain _liti_binary;
retain _trans;

/*********************************************************************
 * Initialization step. Only runs once when starting.
 *********************************************************************/
method init();
  _apply_settings = cat.new_apply_settings();
  _liti_binary = txtanio.new_on_content_server(&liti_binary_path);

  _status = cat.set_apply_model(_apply_settings, _liti_binary);
  if _status NE 0 then put 'ERROR: set_apply_model fails';

  /* Match types are 0=ALL, 1=LONGEST or 2=BEST */
  _status = cat.set_match_type(_apply_settings, 0);
  if _status NE 0 then put 'ERROR: set_match_type fails';

  _status = cat.initialize_concepts(_apply_settings);
  if _status NE 0 then put 'ERROR: initialize_concepts fails';

  _trans = cat.new_transaction();
end;

/*********************************************************************
 * Run step. The method runs per row of input.
 *********************************************************************/
method run();
  set &input_ds(keep=(&document_column &document_id));

  /* Only process if document observation is not empty*/
  if &document_column NE ' ' then do;

    /* Initialize the document with the column data */
    _document = txtanio.new_document_from_string(&document_column);

    /* Set the document on the transaction so we're ready to process */
    _status = cat.set_document(_trans, _document);
    if _status NE 0 then put
      'ERROR: set_document fails on obs:' &document_id;

    /* Apply the binary to the document */
    _status = cat.apply_concepts(_apply_settings, _trans);
    if _status NE 0 then put
      'ERROR: apply_concepts fails on obs:' &document_id;

    /* Look for the concept matches */
    _num_matches = cat.get_number_of_concepts(_trans);
    _i = 0;
```

```
        do while (_i LT _num_matches);
          _name = cat.get_concept_name(_trans, _i);
          _full_path = cat.get_full_path_from_name(_trans, _name);
          _start_offset = cat.get_concept_start_offset(_trans, _i);
          _end_offset = cat.get_concept_end_offset(_trans, _i);
          _term = cat.get_concept(_trans, _i);
          _canonical_form = cat.get_concept_canonical_form(_trans, _i);

          output;

          _i = _i + 1;
        end;

        /* Now look for fact matches */
        _num_matches = cat.get_number_of_facts(_trans);
        _i = 0;
        _canonical_form = '';
        do while (_i LT _num_matches);
          _name = cat.get_fact_name(_trans, _i);
          _full_path = cat.get_full_path_from_name(_trans, _name);
          _start_offset = cat.get_fact_start_offset(_trans, _i);
          _end_offset = cat.get_fact_end_offset(_trans, _i);
          _term = cat.get_fact(_trans, _i);

          output;

          _i = _i + 1;
        end;
        _i = 0;

        /* Clean up resources */
        cat.clean_transaction(_trans);
        txtanio.free_object(_document);
      end;
    end;

    /*************************************************************
     * Termination step that runs only once at the end.
     *************************************************************/
    method term();
      /* clean up resources */
      cat.free_transaction(_trans);
      cat.free_apply_settings(_apply_settings);
      txtanio.free_object(_liti_binary);
    end;
  endthread;
run;

/*************************************************************************
 * Collect output data
 *************************************************************************/
data &output_ds(
  keep=(
    &document_id
    _name
    _full_path
    _start_offset
```

```
      _end_offset
      _term
      _canonical_form
    )
    overwrite=yes
);
dcl THREAD workerth THRD;

method run();
  set from THRD;
end;

enddata;
run; quit;
```

CONCLUSION

SAS Contextual Analysis In-Database Scoring for Hadoop technology provides the SAS user with a powerful framework to deploy and run category, concept, and sentiment scoring models for text analytics in Hadoop. The framework enables you to seamlessly deploy SAS Text Analytics scoring models into a Hadoop cluster. The deployed models are then used to score unstructured text data using SAS DS2. In this paper, we discussed the overall process of SAS Contextual Analysis In-Database Scoring for Hadoop and its basic components. We also demonstrated steps of how a SAS user can modify the DS2 code to score in Hadoop.

REFERENCES

Ghazaleh, David. 2016. "Exploring SAS® Embedded Process Technologies on Hadoop." *Proceedings of the SAS Global 2016 Conference*, Las Vegas, NV: Available at http://support.sas.com/resources/papers/proceedings16/SAS5060-2016.pdf.

Bultman, David. 2016. "Running Text Analytics Models in Hadoop." *Proceedings of the SAS Global 2016 Conference*, Las Vegas, NV: Available at http://support.sas.com/resources/papers/proceedings16/SAS4880-2016.pdf.

ACKNOWLEDGMENTS

David Bultman, Principal Software Developer, Cloud Analytic Services, SAS Institute Inc.

Gail Pearson, Sr. Technical Writer, SAS Institute Inc.

Laura Rock, Manager, Advanced Analytics Testing, SAS Institute Inc.

Peggy James, Sr. Analytics Software Tester, SAS Institute Inc.

RECOMMENDED READING

- *Paper SAS4880-2016, "Running Text Analytics Models in Hadoop" by David Bultman and Adam Pilz, SAS Institute, Inc.*
- *Paper SAS5060-2016, "Exploring SAS® Embedded Process Technologies on Hadoop" by David Ghazaleh, SAS Institute, Inc.*
- *SAS® Contextual Analysis: User's Guide*
- *SAS® Contextual Analysis In-Database Scoring for Hadoop: Administrator's Guide*
- *SAS® Contextual Analysis In-Database Scoring for Hadoop: User's Guide*

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Seung Lee
SAS Institute, Inc.
(919) 531-6659
SeungYong.Lee@sas.com

Xu Yang
SAS Institute, Inc.
(919) 531-3421
Xu.Yang@sas.com

Saratendu Sethi
SAS Institute, Inc.
(919) 531-0597
Saratendu.Sethi@sas.com