SAS® Visual Text Analytics 8.4: User’s Guide

SAS® Documentation
May 21, 2020

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2019. *SAS® Visual Text Analytics 8.4: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Visual Text Analytics 8.4: User's Guide

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

May 2020

8.4-P1:ctxtug

Contents

1

What’s New

What’s New in SAS Visual Text Analytics 8.4

Overview

SAS Visual Text Analytics 8.4 in SAS Viya 3.4 offers new features that enable greater control and customization when modeling and analyzing textual data as well as performance enhancements for some previously existing features.

New Features

The following features are new for SAS Visual Text Analytics 8.4 in SAS Viya 3.4:

- Automatically generate relevant concept rules and fact rules based on existing rules for a concept.

  Note: Automatic concept rule generation is an experimental feature in SAS Visual Text Analytics 8.4.
- Instantaneously add new CLASSIFIER rules to existing concepts by simply highlighting and selecting text in the **Documents** tab.
- Use the sandbox associated with each predefined and custom concept to quickly test new rules and subsets of your model against a document collection.
- Select a pipeline template from the new drop-down list in the New Project window when creating a new project.
- View the breakdown of each topic by sentiment in the Results window for the **Topics** node when a preceding **Sentiment** node is used in the pipeline. In addition, the **Topics** node results include feedback about documents that do not match a topic.
- Kazakh is now supported in addition to 32 other languages. To view the comprehensive list of languages supported, see “Supported Languages” on page 11.
- Stop lists are now provided for the following languages: Arabic, Chinese, Farsi, Japanese, Korean, Tagalog, Thai, and Vietnamese. With these additions, stop lists are now provided for all supported languages.
- Text parsing now supports distributed accumulation. For more information about distributed accumulation, see “Distributed Accumulation” on page 58.
- Analytic store support is now supported for the **Concepts**, **Sentiment**, and **Categories** nodes.

Performance Enhancements

The following performance enhancements are available with SAS Visual Text Analytics 8.4 in SAS Viya 3.4:

- Improved pipeline efficiency. When you run a pipeline after making changes to an analysis node, only the nodes with outstanding changes will be rerun.
- Improved performance when compiling and validating categories.
- Improved performance when compiling and validating concepts.

2 Accessibility

Accessibility in Model Studio 8.4

For information about the accessibility of this product, see Model Studio: Accessibility Features.

3

Introduction to SAS Visual Text Analytics on Viya

What Is SAS Visual Text Analytics in SAS Viya?

Overview

SAS Visual Text Analytics in SAS Viya is a web-based text analytics application that uses context to provide a comprehensive solution to the challenge of identifying and categorizing key textual data. In SAS Visual Text Analytics, you can use the following analysis nodes to build and automate models (based on training documents):

- Concepts
- Text Parsing
- Sentiment
- Topics
- Categories

You can then customize your models in order to realize the value of your text-based data.

Note: Internet Explorer 11 is not supported for SAS Visual Text Analytics 8.4.

SAS Visual Text Analytics in SAS Viya combines the visual programming flow of SAS Text Miner with the rules-based linguistic methods of categorization and concept extraction in SAS Contextual Analysis. These capabilities, along with document-level scoring for each component, are combined in a single user interface.

Using SAS Visual Text Analytics in SAS Viya, you can identify key textual data in your document collections, build concept and categorization models, and remove meaningless textual data.

By default, words that provide little or no informational value (stop words) are excluded from topic analysis. A default stop list is included and automatically applied for several languages. Examples of these words in English include the articles *a*, *an*, and *the* and conjunctions such as *and*, *or*, and *but*. Other terms that are specific to your document collection but provide little or no value due to their low frequency are also identified and excluded. For more information about stop lists, see *Text Mining Action Set: Details* in *SAS Visual Text Analytics 8.4: Programming Guide*.

Visual Text Analytics Basics

SAS Visual Text Analytics provides a number of text analysis nodes that are arranged in a sequence that you control. This sequence takes the form of a pipeline, which empowers you to analyze your document collection with considerable flexibility. When you run a pipeline, the following analyses are performed on data in your project:

- The **Concepts** analysis node in SAS Visual Text Analytics enables you to extract predefined concepts or create additional custom concepts that you can discover in a document or set of documents. For more information about concepts, see “Concepts” on page 6.
- The **Text Parsing** analysis node finds all the terms that are in your document collection. The **Text Parsing** node uses the default stop list provided for the selected project language to determine which terms are excluded from further analysis. In addition, the Text Parsing node displays useful groups of words such as nouns with their modifiers that can be used for topic discovery. For more information about text parsing, see “Text Parsing — Terms and Synonyms” on page 7. For more information about stop lists, see “Start Lists and Stop Lists” on page 8.
- The **Topics** analysis node groups similar documents in a collection into related themes, or *topics*. The documents in each topic often contain similar subject matter, such as motorcycle accidents, computer graphics, or weather patterns. Automatic topic identification enables you to easily categorize each document in your collection. For more information about topics, see “Topics” on page 8.
- The **Sentiment** analysis node determines whether documents express positive, neutral, or negative attitudes. Analysis performed after the Sentiment Analysis node displays a sentiment indicator for each document. For more information about sentiment scoring, see “Sentiment Scoring” on page 9.
- The **Categories** analysis node labels documents based on their content. You can create *categories* using these methods:
  - Specify category (target) variables in your training documents
  - Create new categories that correspond to your organization’s interests
  - Add discovered topics as categories

  For more information about categories, see “Categories” on page 10.

The models that are generated for **Concepts**, **Sentiment**, **Topics**, and **Categories** can then be deployed, and used to automate the process of labeling input documents. You can also register your models, which allows for model governance and model change control over time. For more information about registering models, see “Registering Models” on page 35.

To learn more about each analysis node in detail, continue with the sections below.

Concepts

A *concept* is a property such as a book title, last name, city, gender, and so on. Concepts are useful for analyzing information in context and for extracting useful information (Information Extraction). You can write rules for recognizing concepts that are important to you, thereby creating custom concepts. For example, you can specify that the concept *kitchen* is identified when the terms *refrigerator*, *sink*, and *countertop* are encountered in text.

SAS Visual Text Analytics provides *predefined concepts*, which are concepts whose rules are already written. Predefined concepts save time by providing you with commonly used concepts and their definitions, such as an organization name or a date. You cannot rename predefined concepts, nor can you view or edit their base definitions. You can provide additional rules in the Edit a Concept window to modify or extend their behavior.

The table below shows a list of the predefined concepts for English that are included with SAS Visual Text Analytics, along with their preset priority values. Priority values determine which matches are returned when overlapping matches occur. For predefined concepts and priority values for all languages, see "About Priority Values for Predefined Concepts" on page 148.

Table 3.1 Predefined Concepts and Priorities for English

Predefined Concept	Description	Priority Value
nlpDate	Any date expression (month, day, year, date)	18
nlpMeasure	Measurement or measurement expression (for example, 500kg or 2300 sq ft)	20
nlpMoney	Currency or currency expression	18
nlpNounGroup	Nouns and close modifiers that identify a single object or item (for example, *clinical trial*). Noun groups are typically 2- to 3- word combinations (but can be longer)	15
nlpOrganization	Name of a company or government, legal, or service agency (for example, FBI)	25
nlpPercent	Percentage or percentage expression (for example, 96% or 12 percentage points)	18
nlpPerson	Person's name, including any associated title	20
nlpPlace	Name of a city, country, state, geographical place or region, or political place or region	20
nlpTime	Time or time expression (for example, 6pm or Friday morning)	18

Note: Some languages use a subset of the predefined concepts listed here.

For more information about writing concept rules, see "Writing Concept Rules: Basic LITI Syntax" on page 83.

Text Parsing — Terms and Synonyms

A *parent term* is defined as a label for one or more tokens that represent a grouping of variants (one or more surface forms) that are related, as defined by underlying rules or algorithms. In SAS Visual Text Analytics, a term is the basic building block for topics, term maps, and category rules. Each term has an associated role that either is blank or identifies that term's part of speech. A *surface form* is a variant of a parent term that is located in a matched subset of text. Surface forms can include inflected forms, synonyms, misspellings, and other ways of referring to a parent term. SAS Visual Text Analytics can identify and classify misspellings of terms based on similarity and frequency. Because misspellings actually refer to another term, they are treated as synonyms during analysis.

A *synonym list* is a way for users to create custom parent terms or to add terms grouped under a parent term. It is a SAS data set that identifies pairs of words that should be combined as single terms for the purposes of analysis. Synonyms are applied at the parent level; all variants of each parent term are combined together into one group. You can specify a synonym list in the **Text Parsing** node. Synonym lists are stored in data sets and have a required format.

In SAS Visual Text Analytics 8.3 and previous releases, the terms in a synonym list were applied to only child terms when using distributed accumulation. Therefore, the child term in a synonym list corresponded to the child term in the **Terms** table.

In SAS Visual Text Analytics 8.4, if the child term in a synonym list is the same role as a parent term in the **Terms** table, then the parent term and all its children terms in the **Terms** table will appear under the parent term in the synonym list. If a child term in the synonym list matches a child term in the **Terms** table, then only the child term in the **Terms** table appears under the parent term in the synonym list.

Note: If a synonym list includes multiple entries that assign the same terms to different parents, then the parsing results reflect only the first entry.

The synonym list must include the following variables:

- TERM, which contains a term to treat as a synonym of the PARENT.
- PARENT, which contains the representative term (label) to which the TERM should be assigned.
- TERMROLE, which enables you to specify that the synonym is assigned only when the TERM occurs in the role specified in this variable. A *term role* is a function performed by a term in a particular context; term roles include part-of-speech roles, entity roles, and user-defined roles. Users can define these roles in the Concepts node. In order for the user-defined roles to be available in the Text Parsing node, the Concepts node needs to precede it in the pipeline. TERMROLE can also have an empty value.

You can also include the variable PARENTROLE, which enables you to specify the role of the PARENT.

Note: SAS Visual Text Analytics 8.4 requires that a role is provided for each term in the synonym list that has more than one role in the terms list. If a role is not provided for each term in the synonym list that has more than one role in the terms list, you could encounter an error that will cause processing to stop.

Start Lists and Stop Lists

You use start lists and stop lists to control which terms are kept or dropped during text parsing. The parsing results also control the terms that are used in topic discovery. A *start list* is a data set that contains a list of terms to include in the parsing results. If you use a start list, then only terms that are included in that list appear in parsing results. A *stop list* is a data set that contains a list of terms to exclude from the parsing results. You can use stop lists to exclude terms that contain little information or that are extraneous to your text mining tasks. A default stop list is provided for all supported languages in SAS Visual Text Analytics 8.4 in the library *ReferenceData*.

Start lists and stop lists have the same required format. You must include the variable TERM, which contains the terms to include (start) or exclude (stop). You can also include the variable ROLE, which contains an associated role. If you specify a ROLE variable, then terms are kept (for a start list) or dropped (for a stop list) only if their role is the one that is specified in the ROLE variable.

Topics

Topics are derived from natural groupings of important terms that occur in your documents. In SAS Visual Text Analytics, topics are automatically generated and assigned to documents. A single document can contain more than one topic.

The interactive window for the Topics node displays all the topics that SAS Visual Text Analytics identified. The default name of a topic is the top five terms that appear frequently in the topic. These terms are sorted in descending order based on their weight.

Sentiment Scoring

Sentiment analysis is the process of identifying the author's tone or attitude (positive, negative, or neutral) expressed in a document. SAS Visual Text Analytics uses a set of proprietary rules that identify and analyze terms, phrases, and character strings that imply sentiment. A sentiment score is then assigned, based on that analysis. Using these rules, the software is able to provide repeatable, high quality results.

The assignment of sentiment to a document is based on the attitude that is associated with the document as a whole. For example, the following document would have a positive sentiment: `Had an awesome time yesterday. Glad I bought my tent from Store XYZ.`

Because documents can be associated with multiple words or terms that imply sentiment, SAS Visual Text Analytics uses a scoring system to assign a final sentiment score. Below is the list of languages that have the officially supported base sentiment model:

- Arabic
- Chinese (Simp./Trad.)
- Dutch
- English
- Farsi
- French
- German
- Italian
- Japanese
- Korean
- Portuguese
- Spanish
- Turkish

If a sentiment model does not exist for the project language, the following message appears: `No default sentiment model exists of the language 'PROJECT LANGUAGE'.` The Sentiment node runs without errors, but it does not produce any results. However, you can upload your own sentiment model in SAS Visual Text Analytics.

The following list provides basic information about how sentiment scoring works. (The information has been simplified to illustrate key concepts.)

- Each positive term or phrase is worth a single (positive) point.
- Each negative term or phrase is worth a negative point.
- If there are more positive terms or phrases than negative, the final sentiment score is positive.
- If there are more negative terms or phrases, the final sentiment score is negative.
- If there are an equal number of positive and negative terms or phrases, the sentiment score is neutral.

The formulas used in calculating each sentiment score is shown below:

$$\mathbf{RawScore(object)} = (\sum_{pos} rule\_weight * pos\_to\_neg\_ratio \; - \sum_{neg} rule\_weight\,)$$

$$\mathbf{CummulativeScore(object)} \\ = RawScore(object) + \Sigma_{childs}\lambda_{child} * CummulativeScore(child)$$

$$\mathbf{PositiveProb(object)} = sigmoid(CummulativeScore(object))$$

$$\text{Where } \mathbf{sigmoid}(\mathrm{x}) = \frac{1}{1+e^{-x*\ln(1.5)}}$$

Definitions for the formulas above are as follows:

- Lambda is the weight of the corresponding node.
- The value *rule_weight* is the weight of the individual sentiment rule.
- The value *object* refers to a node in the taxonomy, and the document itself is considered as the topmost node in the taxonomy.

Categories

A *category* identifies a group of documents that share a common characteristic. For example, you could use categories to identify the following:

- areas of complaints for hotel stays
- themes in abstracts of published articles
- recurring problems in a warranty call center

You can create a category using one of the following methods:

- Add a topic as a category
- Specify a category variable
- Create a new category in the interactive window for the Categories node

The Categories node cannot process unary categories. You can edit the rules that are automatically generated for category variables and for topics that are added as categories. You can also write your own rules for custom categories.

Note: The category rules are in the format that SAS Visual Text Analytics uses (MCAT), rather than in LITI format. You can refer to LITI concepts from within categories.

For information about writing category rules, see "Writing Category Rules: Boolean Rules" on page 102.

Using Taxonomies

In SAS Visual Text Analytics, you can create category and concept rule sets, which are organized into a taxonomic structure. Each taxonomy consists of *tree nodes* (not to be confused with analysis nodes). Each tree node is a container for one or more rules. The taxonomy is used to organize rules and reflect the overall model design and to make testing, refinement, and maintenance of rules easier. Rules can explicitly reference other tree nodes, but there are no implied dependencies within the tree that impact results (like dependencies of inheritance).

Concept and category taxonomy trees can be organized in any way that is useful for your objectives. However, using a careful and principled design process is recommended for larger projects. For example, commonly referenced rules should be placed in a location where they are easy to find and their shared status is apparent. Naming concept or category tree nodes should enable easy navigation among nodes. For information about naming conventions, see *Create a custom concept* in "Considerations when Creating a Custom Concept" on page 40. Each category node in the tree is a container for a single rule. By contrast, under a concept node, there can exist multiple rules.

Supported Languages

SAS Visual Text Analytics 8.4 supports the following languages. To license additional languages, See your SAS sales representative.

Table 3.2 SAS Visual Text Analytics 8.4 Supported Languages

Arabic
Chinese (Simp./Trad.)
Croatian
Czech
Danish
Dutch
English
Farsi
Finnish
French
German
Greek
Hebrew
Hindi
Hungarian
Indonesian
Italian
Japanese
Kazakh
Korean
Norwegian (Bok./Nyn.)

Polish
Portuguese
Romanian
Russian
Slovak
Slovene
Spanish
Swedish
Tagalog
Thai
Turkish
Vietnamese

4

Managing Projects

Getting Started

Preparing the Document Collection

Before you create a project in SAS Visual Text Analytics, you need to prepare your document collection for analysis. SAS Visual Text Analytics enables you to analyze document collections stored in various formats. For a complete list of supported data formats, see “Making Data Available to CAS” in *SAS Data Explorer: User’s Guide*. You can select a data source and then identify the text variable that you want to analyze. You also have the option to select category variables for analysis.

When you prepare the input document collection, you should select a set of documents that is representative of the documents that you want to process later. The terms and patterns that exist in the input document collection influence the creation of any models.

Your priorities for the creation of the input document collection depend on the specific goals of your Text Analytics project. However, the following guidelines can help you prepare your input document collection:

- For categorization projects, you should include at least 200–400 documents for each category that you want to target.
- For complex categories, a collection of 2000–3000 documents for each category that you want to target is ideal.

- In order to take advantage of interactive visual displays, reduce the size of very large document collections. Very large collections take a longer time to render in term maps, for example.

In SAS Visual Text Analytics 8.4, you can import documents that are larger than 100 KB. However, importing an extremely large document can result in a data-loading error, and can cause trouble when viewing the data table. Also, extremely large documents can lead to slower performance in an interactive window, as well as truncation of information in a documents table.

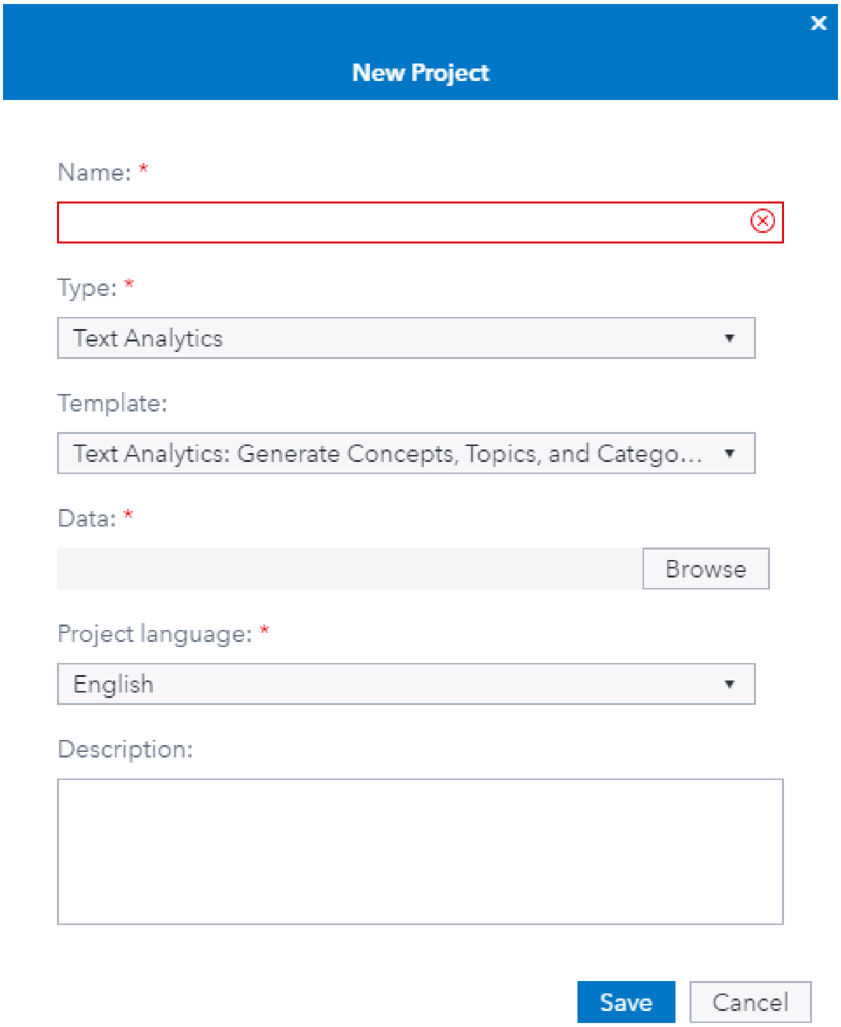
For document collections that are not prepared for analysis, you can leverage the document conversion feature in the Browse Data window. For more information about document conversion, see "Overview of Document Conversion" in *SAS Data Explorer: User's Guide*.

Note: The input CAS table should not contain a variable named `__uniqueid__`. SAS Visual Text Analytics generates a `__uniqueid__` variable during project creation, and having a duplicate of this variable can result in an error.

Creating a Project

To create a project in **Model Studio**, complete the following steps:

1 Navigate to the **Projects** page, and click **New Project** in the upper right corner of the page. The New Project window appears.



New Project

Name: *

Type: * Text Analytics

Template: Text Analytics: Generate Concepts, Topics, and Catego…

Data: *

Project language: * English

Description:

2 Enter a project name in the **Name** field.

3. Select **Text Analytics** from the drop-down list in the **Type** field.
4. Select a pipeline template from the drop-down list in the **Template** field.
5. Click the **Browse** button in the **Data** field to open the **Choose Data** window. Select the data source that you want to use, and click **OK**.
6. Select a project language from the drop-down list in the **Project language** field. For a comprehensive list of the languages that are supported in SAS Visual Text Analytics 8.4, see “Supported Languages” on page 11.
7. Click **Save** in the lower right corner of the New Project window.

After you create your new project, Model Studio takes you to the **Data** tab. Here, you can make adjustments to data source variable type and role. Once a project is created, any changes that you make to it are automatically saved. For more information about the **Data** tab, see “Assigning Variables in the Data Tab” on page 15.

Assigning Variables in the Data Tab

Once a project has been created, double click on the project to open it. The **Data** tab displays the variables in the data set, the variable type (Numeric or Character) of each variable, each variable's role (Category, Text, or Key), and display status (Yes or No). Model Studio requires you to assign the role of Text to one variable in your data set. To assign variable roles, complete the following steps:

1. Select a variable from the **Variable Name** column.
2. In the upper right corner of the **Data** tab, select the desired role from the drop-down list under **Role**.
3. Once a role is selected, it is automatically assigned to the selected variable.

Variables that are assigned the role of Category appear in the **Categories** pane when the option **Automatically generate categories and rules** is selected. For more information about categories, see “Using the Interactive Window for the Categories Node” on page 70. Display variables become columns in the **Documents** tab of all pipeline nodes with the exception of the **Data** node and **Sentiment** node. To change the display status of variables, click the check box to the left of each variable that you want to modify. Once variables have been selected, click the check box next to **Display variable** in the upper right corner of the **Data** tab. The display status of the selected variable or variables changes instantly.

Note: As long as a variable is assigned the role of **Text** , it acts as a display variable. However, you can choose whether to display variables assigned the role of **Category**.

Changing the Data Source

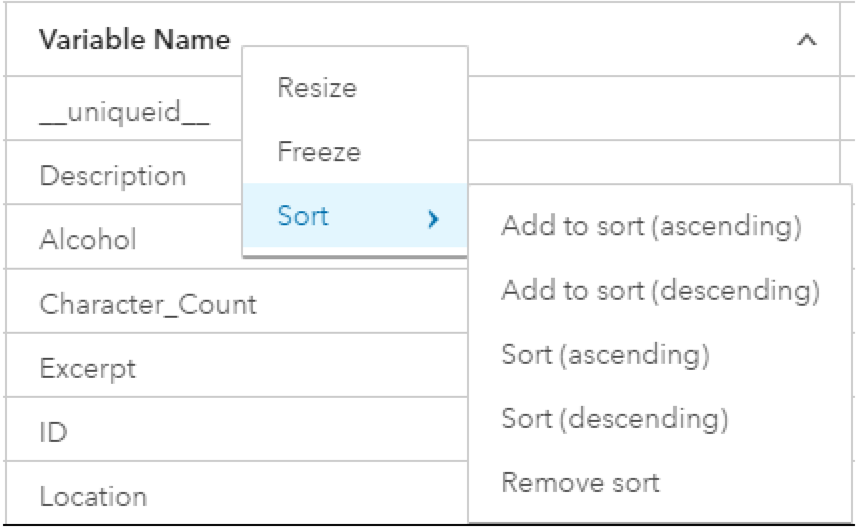
After you run a pipeline, you can change the data source without having to create a new project. To change the data source, navigate to the **Data** tab. In the upper left corner of the **Data** tab, click . The Browse Data window appears, and you can select a new data source. Once you select a new data source, click **OK** to begin the process of replacing the original data source. After you replace a data source, you can assign variable roles and run your pipeline.

Note: Replacing a data source does not affect node settings.

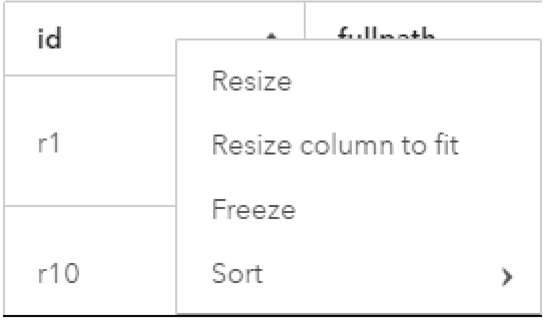
Customizing Views in the Data Tab

In the Data Tab, there are two different ways of viewing the information present. The default view in the Data Tab shows the **Variables table**, which has columns for **Variable Name**, **Type**, **Role**, and **Display Variable**. The second option for viewing information about the data set being used is the **View table** option. To switch from the **Variables table** to the **View table**, click the icon in the upper left corner of the **Data** tab, next to the filter bar. The **View table** shows greater detail, and has a column for each of the variables in the data set.

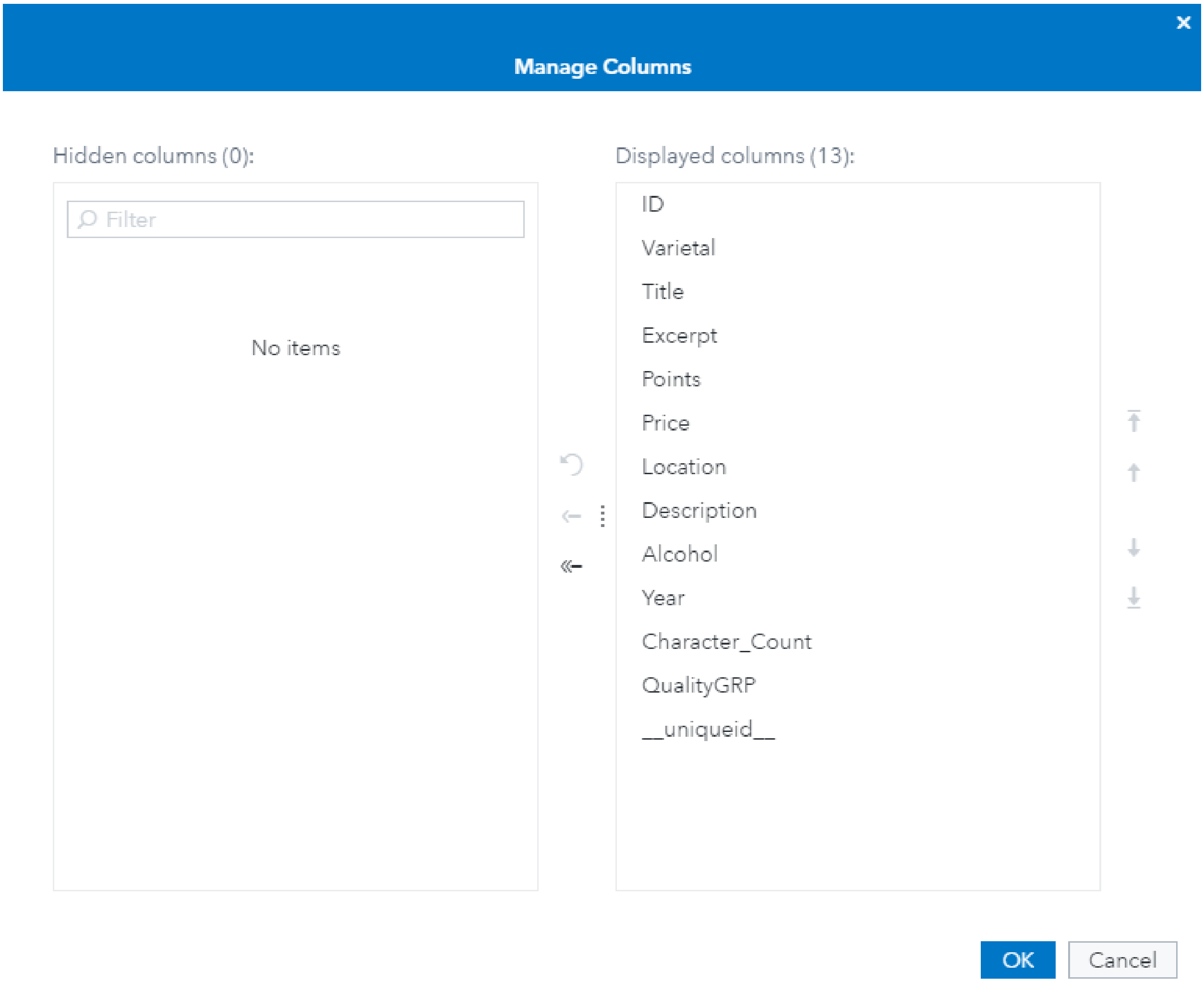
To customize your view in the **Variables table** , you can right-click on column headings to resize, sort, or freeze a column.



You can also customize your view in the **View table**, which contains a similar set of options. However, the **View table** also offers a **Resize column to fit** option.



Resizing columns is advantageous when there are lengthy documents in your collection, as it enables you to see, add, or discard columns in the **Manage columns** window, which is made available by clicking the ⋮ icon in the top right corner of the **Data** tab. In the window, a list of **Hidden columns** and a list of **Displayed columns** are shown.



Using the icons between the two lists, you can move variables from the **Displayed columns** list to the **Hidden columns** list, and from the **Hidden columns** list to the **Displayed columns** list.

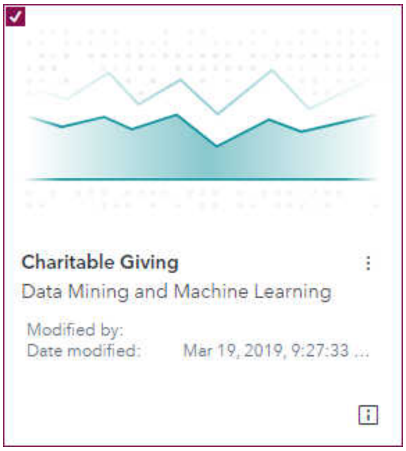
Sharing a Project

After creating a project, you can share it with others in your organization. Model Studio enables you to share projects with user-defined groups.

The Model Studio implementation of sharing is distinct from project sharing as performed in SAS Drive. Any projects that you share using SAS Drive do not retain the same settings for user groups in Model Studio. Also, any projects that you share using Model Studio do not retain the same settings for users in SAS Drive. For more information about the authorization service, see SAS Viya Administration: General Authorization. For more information about SAS Drive, see SAS Drive: Getting Started.

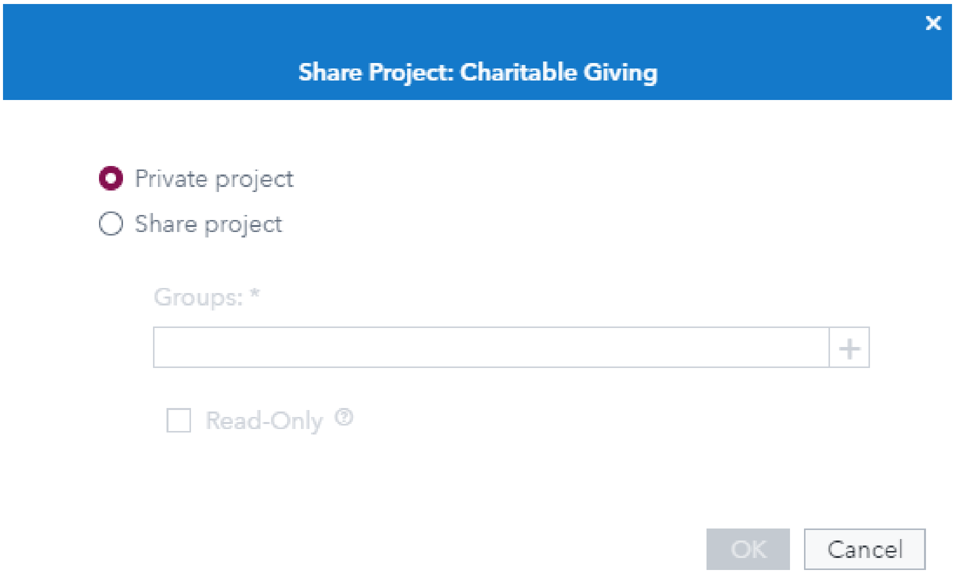
To share a project:

1 Select the desired project by clicking the check box in the project tile, and then click the ⋮ icon next to the Project page Toolbox.


Charitable Giving
Data Mining and Machine Learning
Modified by:
Date modified: Mar 19, 2019, 9:27:33 ...

2 Select **Share**.

3 The Share Project window appears.


Share Project: Charitable Giving

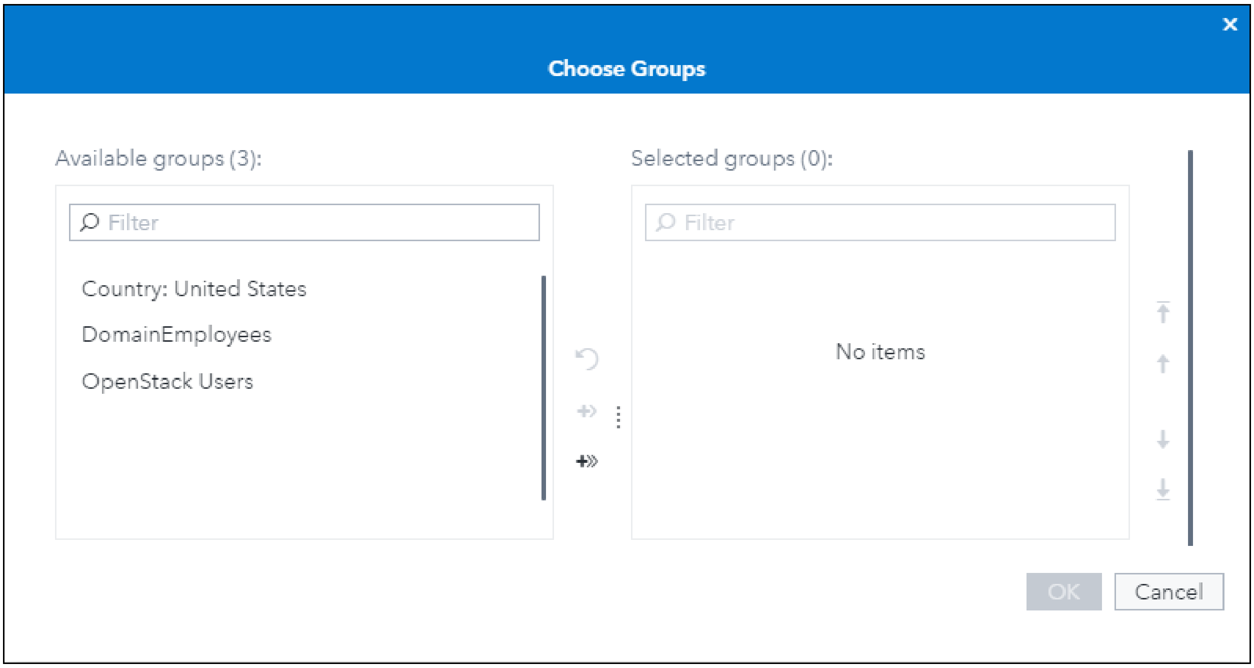
● Private project
○ Share project

Groups: *

□ Read-Only

4 Select **Share project**.

5 Configure the groups by clicking the + icon. Use the Choose Groups window to select which groups you want to share access with.



Choose Groups

Available groups (3):

Country: United States

DomainEmployees

OpenStack Users

Selected groups (0):

No items

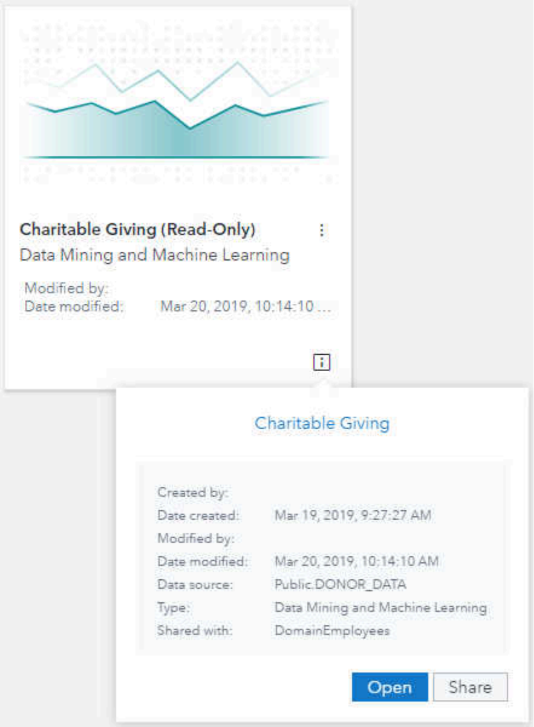
Once groups have been configured, click **OK**.

6 By default, group members can modify the shared project. To disable this feature, select **Read-Only**.

Note: The following features apply to shared projects:

- Only the owner of a shared project can change shared status of that project.
- Only the owner of a shared project can delete that project.
- If a project is not shared in **Read-Only** mode, then only one person can have the project open at a time. Shared projects that are currently open are indicated with a 🔒 icon on the Projects page.
- If a project is shared in **Read-Only** mode, nobody can make changes to the project, including the project owner.
- SAS Administrators must be included in any group that the project is shared with.

7 Once the configurations are set on the Share Project window, click **OK** to share. You can see that your project has been shared on the project tile.



Charitable Giving (Read-Only)
Data Mining and Machine Learning

Modified by:
Date modified: Mar 20, 2019, 10:14:10 ...

Charitable Giving

Created by:	
Date created:	Mar 19, 2019, 9:27:27 AM
Modified by:	
Date modified:	Mar 20, 2019, 10:14:10 AM
Data source:	Public.DONOR_DATA
Type:	Data Mining and Machine Learning
Shared with:	DomainEmployees

You can also remove sharing of a project. To do this, repeat steps 1 through 3 above, but in the Share Project window select **Private project** and click **OK**. This removes shared access to the project.

Promotions and Upgrades within SAS Viya

Promotions Considerations

A *promotion* is the process of making resources that exist in one environment present, available, and usable in another environment. The promotion process consists of exporting the resources from the source environment and then importing the resources to the target environment. For more information about promotions, see Promotion: Overview .

Consider the following information before performing a promotion:

- The owner of a project that is being promoted must sign in to the target environment before any projects can be imported. If you are a project owner, it is recommended that you promote your own individual projects.
- Before you promote a project, you must promote the input data for that project to the target environment.
- A user-created pipeline or node template must be promoted separately before you can use it in a new project. If a custom template is derived from another template in The Exchange, both templates are required to be on the source system in order to successfully import projects that use the template.
- You must rerun all nodes and pipelines in a promoted project before the results are available on the target environment.

Promotions from Model Studio 8.2 to Model Studio 8.3 and Later

When promoting a project from Model Studio 8.2 to a newer version of Model Studio, consider the following information:

- Before promoting a project from Model Studio 8.2, you must first apply the latest software update on the Model Studio 8.2 server.
- When promoting a project from a Model Studio 8.2 system to a different Model Studio environment, you must promote any templates used by that project to the target environment. Both projects and templates must be promoted using the CLI (Command Line Interface) in this scenario. Instructions for using the CLI to promote projects and templates can be found in "Promotion within SAS Viya: Instructions" in *SAS Viya Administration: Promotion (Import and Export)*.

Note: When upgrading from Model Studio 8.2 to Model Studio 8.3 and beyond within the same environment, project tiles are not displayed appropriately until the project is manually upgraded.

Promotions from Model Studio 8.3 to Model Studio 8.3 and Later

You can promote projects, pipeline templates, and node templates from Model Studio 8.3 to Model Studio 8.3 and later. Before promoting your content from Model Studio 8.3, consider the following information:

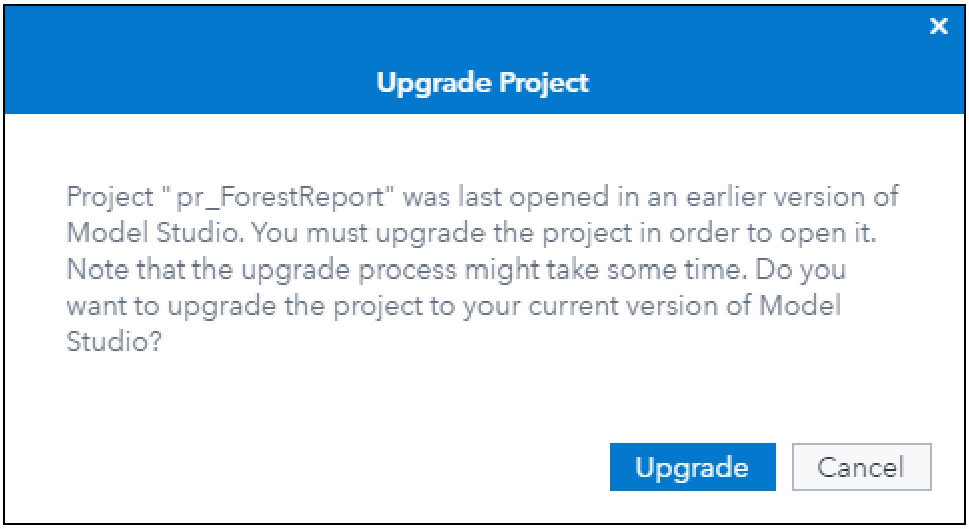
- If you need to quickly promote a single project within the same version of Model Studio, use the instructions in "Importing and Exporting a Project" on page 23 to export the project from the source environment and import the project in the target environment.
- To promote a project or template from Model Studio 8.3 to Model Studio 8.3 or later, you can follow either the CLI instructions or the Wizard instructions found in "Promotion within SAS Viya: Reference" in *SAS Viya Administration: Promotion (Import and Export)*.

Upgrade Considerations

An *upgrade* to Model Studio adds significant feature changes or improvements to the product.

Consider the following information before performing an upgrade:

- If you are upgrading Model Studio within the same version of SAS Viya, see "Adding SAS Viya Software to a Deployment and Upgrading Products in SAS Viya 3.4" in *SAS Viya for Linux: Deployment Guide* for more information.
- If you are upgrading Model Studio in addition to upgrading SAS Viya, see "Upgrading to SAS Viya 3.4 from Earlier Versions of SAS Viya" in *SAS Viya for Linux: Deployment Guide* for more information.
- After all the steps have been completed in the *SAS Viya for Linux: Deployment Guide* and Model Studio or SAS Viya has been upgraded, users can upgrade their individual projects. To upgrade a project:
  - Sign in to Model Studio. The ⊘ icon in the lower left corner of the project tile indicates that the project has not been upgraded.
  - Open the project that you want to upgrade, and click the **Upgrade** button in the Upgrade Project window.



Upgrade Project

Project " pr_ForestReport" was last opened in an earlier version of Model Studio. You must upgrade the project in order to open it. Note that the upgrade process might take some time. Do you want to upgrade the project to your current version of Model Studio?

- When a shared project is upgraded, it becomes a private project. After you upgrade a project, you must re-share it. It is recommended that you take note of all your shared projects, and with whom they are shared, before upgrading.
- If you are the project owner, you must upgrade the projects that you created. SAS Administrators cannot upgrade projects that are created by other users.
- Before you upgrade a project, you must load the input data for that project to the target environment.
- After your project is upgraded and you run your pipelines, the models in the project are no longer registered. You must re-register and re-publish your models.

Promotion Considerations Specific to SAS Visual Text Analytics

When promoting a SAS Visual Text Analytics project, consider the following information:

- Before promoting a project, run all pipelines that are used by that project to ensure that custom components are preserved. For example, topics that are added as categories will not be preserved after a promotion if this action is not completed.
- When you promote a project that uses a custom start list, stop list, synonym list, or sentiment model, you must promote those custom components to the target environment before promoting the project. Otherwise, pipelines containing nodes that use those components will fail in the target environment.
- If promoting a project that contains user-specified category variables to a more recent version of SAS Visual Text Analytics, run the pipeline that contains that Category node after the project has been promoted. Otherwise, the rules created for automatically generated categories reflect those from the older SAS Visual Text Analytics environment instead of those generated within the new SAS Visual Text Analytics environment.
- Import of categories and concepts whose names contain colons will fail if transferring a SAS Visual Text Analytics project to a newer version of SAS Visual Text Analyticsin a separate environment. However, if you promote a project to a newer version of SAS Visual Text Analytics within the same environment, projects containing categories or concepts with invalid names do upgrade successfully.
- Topics that are created by merging two topics are preserved when upgrading to a newer SAS Visual Text Analytics environment, but do not produce document matches. To generate matches for a merged topic, re-create the topic in the new environment and rerun the pipeline.

If you are upgrading to a newer SAS Visual Text Analytics environment, or transferring a project from an older SAS Visual Text Analytics environment to a newer environment, complete the following steps. These steps enable you to preserve user-defined topics when promoting a project.

1. Apply the latest software update on the server hosting the older SAS Visual Text Analytics environment.

2. Open the project containing the user-defined topics.

3. After opening the project containing the user-defined topics, you can proceed with your upgrade to the newer SAS Visual Text Analytics environment.

Importing and Exporting a Project

To import or export a project, you must belong to the SAS Admin group. Only projects that were created using SAS Visual Text Analytics 8.4 can be imported into a SAS Visual Text Analytics 8.4 environment. To import a project that was created in an earlier version of SAS Visual Text Analytics, you must follow the instructions provided in “Promotions and Upgrades within SAS Viya” on page 20. To export a project, complete the following steps:

1. On the Projects page, select the project that you want to export.

2. Click the ⋮ icon and select **Export**.

The project files will immediately begin to download. SAS Visual Text Analytics projects are stored as JSON files.

Note: JSON files are saved in a ZIP file that you specify when exporting a project.

In order to import a project, complete the following steps:

1. Click the ⋮ icon and select **Import**. If you also have SAS Visual Forecasting or SAS Visual Data Mining and Machine Learning installed, select **Import** ⇨ **Visual Text Analytics**.

2. In the Import Text Analytics Project window, specify the location of the project and an associated data set. When you import a project, you must specify the ZIP file that was saved when you exported the original project.

3. Click **Import**.

Importing a SAS Contextual Analysis Project

Overview

SAS Visual Text Analytics offers the ability to import a SAS Contextual Analysis project. With the click of a button, users can import concepts and categories from an existing SAS Contextual Analysis project into SAS Visual Text Analytics. This eliminates the need to manually redefine rules and taxonomies that were created in a SAS Contextual Analysis project. When importing a project from SAS Contextual Analysis, keep the following in mind:

- Projects that have more than 3000 categories or concepts can result in an error during the import process. Increasing Java heap size reduces the chance that an error will occur when importing projects with large taxonomies.

- Some part-of-speech tags are mapped from SAS Contextual Analysis to SAS Visual Text Analytics, and can surface differently depending on the mapping.

When a project is imported from SAS Contextual Analysis, SAS Visual Text Analytics imports the following:

- Predefined Concepts
- Custom Concepts
- Custom Categories

Predefined concepts can surface in different ways when a project created in SAS Contextual Analysis is imported in SAS Visual Text Analytics. SAS Visual Text Analytics does not support all imported predefined concepts, such as TIME_PERIOD. Imported predefined concepts that are not supported in SAS Visual Text Analytics still appear in the **Custom Concepts** list. However, no rule is generated for that concept, which means you have to write the rules for it.

Imported predefined concepts that are supported in SAS Visual Text Analytics appear in the **Predefined Concepts** list. If custom rules were added to a supported predefined concept in SAS Contextual Analysis, they are shown in the **Edit a Concept** panel when that concept is selected.

There are some predefined concepts that are not supported in SAS Visual Text Analytics for which rules are still created. These predefined concepts are placed in the **Custom Concepts** list, and their rules point to hidden definitions in order to enhance backward compatibility. These rules are preceded by a lowercase `s` to signify that the rule points to a hidden SAS concept. Here is an example of what a rule for this type of predefined concept would look like: `CONCEPT:sAddress`.

Note: Concepts are imported only if the name of the concept is restricted to single-byte letters, numbers, and underscores.

When you import a SAS Contextual Analysis project, the categories contained within that project are imported along with it. There are differences in the ways that categories are surfaced in SAS Visual Text Analytics relative to the ways that they appear in SAS Contextual Analysis.

One of the most noticeable differences is the way that SAS Visual Text Analytics counts the number of documents that contain matches for a category. Matches found for a child category are rolled into the number of matches found for a parent category in SAS Contextual Analysis. However, the matches found for a child category in SAS Visual Text Analytics are not rolled up to their parent category. This means that you might see fewer matches in SAS Visual Text Analytics than in SAS Contextual Analysis for the same category.

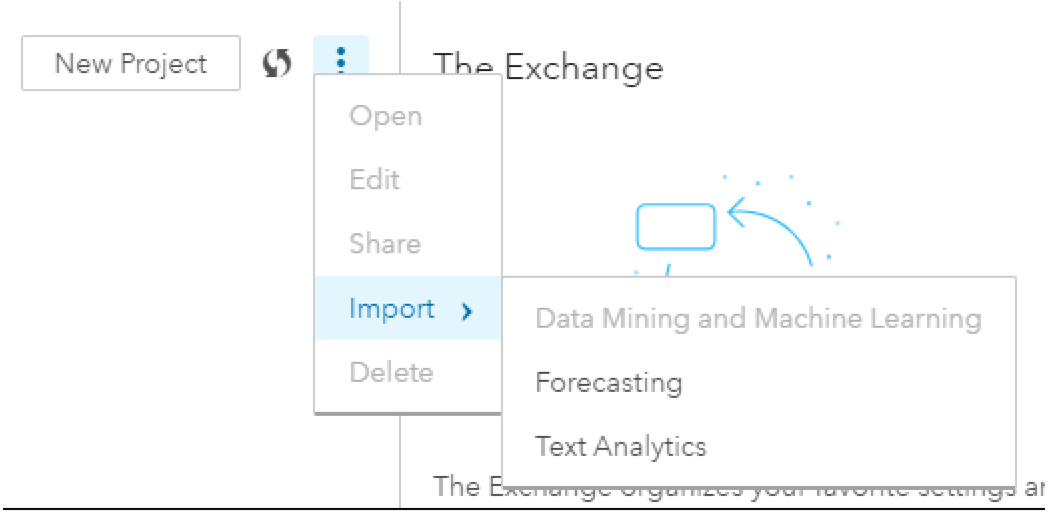
TIP When you import a category with a SAS Contextual Analysis project, consider the following:

- If you create a category variable, ensure that you did not use that same variable as a category variable in the project that you are importing. This results in an error if you select the **Automatically generate categories and rules** option.
- Categories that are created in a SAS Contextual Analysis project are imported and displayed under **All Categories** in the **Categories** panel.

How do I Import a Project from SAS Contextual Analysis?

On the Projects page in SAS Visual Text Analytics, you can import a project from SAS Contextual Analysis. In order to import a project from SAS Contextual Analysis, complete the following steps:

1. In the upper right corner of the Projects page, select the ⋮ icon.
2. Select **Import**. If you have SAS Visual Forecasting or SAS Visual Data Mining and Machine Learning licensed in addition to SAS Visual Text Analytics, be sure to select **Text Analytics**.


New Project
The Exchange
Open
Edit
Share
Import ›
Delete
Data Mining and Machine Learning
Forecasting
Text Analytics
The Exchange organizes your favorite settings ar

3 In the Import Text Analytics Project window, click the **Browse** button next to the **File (json):** field and select the project that you want to import. You are returned to the Import Text Analytics Project window.

4 Name your project in the **Name:** field.

5 Select a project language from the drop-down menu in the **Language:** field.

6 Select a data source using the **Browse** function for the **Date source:** field, and click **OK**. This returns you to the Import Text Analytics Project window.

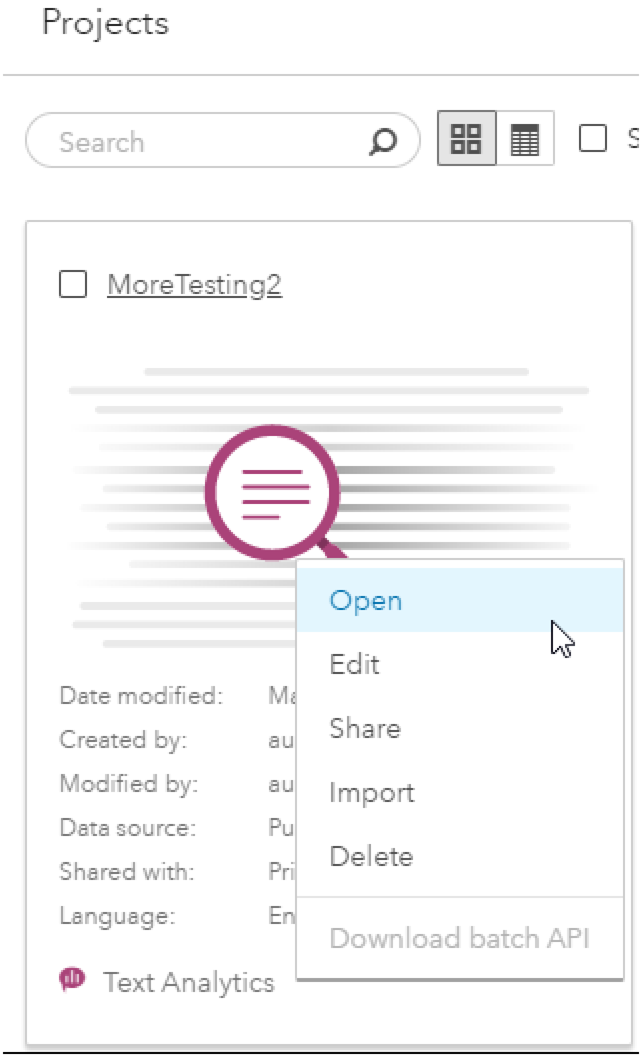
7 Click **Import** in the bottom right corner of the Import Text Analytics Project window.

Note: When importing a project, there is no indication that the import is taking place. However, once the import is complete, the project is shown on the Projects page.

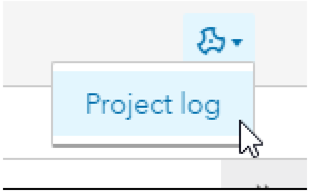
When a project is successfully imported, users can view the Project Import Log for details about the following:

- Creation of categories
- Creation of concepts
- Location of imported predefined concepts
- Errors that occurred during the import process

In order to view the Project Import Log, open the imported project from the Projects window.



This brings you to the **Data** tab, which is located within the project window. In the upper right corner of the project window, click the ⚙ icon and select **Project log**.



The project log appears, showing all errors, warnings, and notes associated with the project. When you finish reviewing the Project Import Log, click **Close** in the lower right corner of the window.

5

Working With Pipelines

Overview of Pipelines

Model Studio projects are built around one or more pipelines. A *pipeline* is a process flow diagram that can be used to represent a sequence of analytical tasks. These analytical tasks are represented as individual nodes in a pipeline.

By default, the initial pipeline for a project uses the template that was specified when the project was created. You can create new pipelines using different templates, and you can make changes to the initial pipeline.

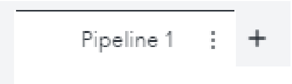
Creating a New Pipeline

In Model Studio, pipelines contain the nodes that process data and create models. A project can contain multiple pipelines.

To create a new pipeline:

1 Navigate to the **Pipelines** tab.

2 Click the + icon next to the current pipeline tab in the upper left corner of the canvas.



The New Pipeline window appears.

3. Give the pipeline a name and an optional description.

4. In the **Template** field, your recently used templates are available. To use a template that you have not used recently, select **Browse templates** and select a template in the Browse Templates window.

5. Click **Save**.

You can also duplicate a pipeline. Click the ⋮ icon next the current pipeline tab in the upper left corner of the canvas and click **Duplicate**.

Note: The duplicate functionality is not available in SAS Visual Text Analytics 8.4.

Actions on the Pipeline

Click the ⋮ icon on the current pipeline tab in the upper left corner of the canvas to perform the following actions:

- **Run** — Runs the entire pipeline.
- **Stop** — Stops the run when the pipeline is running.
- **Duplicate** — Creates a duplicate pipeline. The name is appended with a number. You can rename the duplicate after it is created.

  Note: The duplicate functionality is not available in SAS Visual Text Analytics 8.4.
- **Rename** — Renames the pipeline.
- **Save to The Exchange** — Saves the pipeline with the nodes and any settings applied to those nodes as a template to The Exchange. The new templates can be used in other projects.
- **Delete** — Deletes the pipeline. This option is available when you have more than one pipeline in your project.
- **Show overview map** — Places a map of the pipeline in the upper left corner of the canvas.
- **Expand header** — Provides a space at the top of the tab to add a description or other text that might be useful. The text can be formatted.

Modifying a Text Analytics Pipeline

Pipelines are flexible. You can create additional pipelines or modify the default pipeline by adding different nodes. The different nodes within a pipeline are organized into groupings of nodes that share similar characteristics, and are visually grouped by color. The pipeline groupings in SAS Visual Text Analytics are as follows:

1. Natural Language Processing, which includes the **Concepts** and **Text Parsing** nodes.
2. Feature Extraction, which includes the **Topics** node.
3. Text Modeling, which includes the **Categories** node.

4 Miscellaneous, which includes the **Sentiment** node.

When you add a node to a pipeline, a set of governing rules is applied to ensure the proper ordering of the nodes. If a node is upstream relative to another node, then it is a *parent node*. If a node is downstream relative to another node, then it is a *child node*. Some nodes cannot be created without the appropriate parent or child node. For example, a **Topics** node requires that a **Text Parsing** node precedes it. If such a predecessor does not exist, then the governing rules prevent the inclusion of a Topics node. In order to add a new parent node to a pipeline, complete the following steps:

1 Navigate to the pipeline view.

2 Right-click on an existing node in the pipeline, and select **Add parent node** from the pop-up menu.

3 Select the desired node type from the resulting pop-up menu.

In order to add a new child node to a pipeline, complete the following steps:

1 Navigate to the pipeline view.

2 Right-click on an existing node in the pipeline, and select **Add child node** from the pop-up menu.

3 Select the desired node type from the resulting pop-up menu.

Where applicable, the output of a given node is used within (flows into) its successors. Here are some examples:

- When one or more Concepts nodes precede a Text Parsing node, the Text Parsing node uses the concepts from all its predecessor nodes during text parsing and extracts relevant terms.
- When a Text Parsing node precedes a Concepts or Categories node, all the kept terms from the Text Parsing node are included in the concepts and categories interactive views as textual elements. These textual elements can be used to develop rules for concept extraction or categorization.
- When a Topics node precedes a Categories node, you can select one or more topics in the Topics interactive window and add them as categories. These categories and the associated category rules are automatically created when any of the succeeding Category nodes run.
- Within the rules in a Categories interactive window, you can refer to concepts defined in the preceding Concepts node. For more information about referring to concepts in categorization rules, see “Introduction to Category Rules” on page 102.
- Within the interactive views that follow a Sentiment node, the document level sentiment information is shown alongside the document text.

Overview of Templates

Model Studio supports templates as a method for creating statistical models quickly. A *template* is a special type of pipeline that is pre-populated with configurations that can be used to create a model. A template might consist of multiple nodes or a single node. Model Studio includes a set of templates that represent frequent use cases, but you can also create models themselves and save them as templates in the toolkit.

Creating a New Template from a Pipeline

To create a template from a pipeline:

1 Click the ⋮ icon next to the pipeline tab in the upper left corner of the canvas.

2 Select **Save to The Exchange**.

3. In the Save Pipeline to The Exchange window, enter a **Name** and **Description** for the new template.
4. Click **Save**.

You can also create templates from singular nodes. To create a template from a node:

1. Right-click on the desired node. Select **Save As**. The Save Node to The Exchange window appears.
2. In the Save Node to The Exchange window, enter a **Name** and **Description** for the new template.
3. Click **Save**.

Running a Pipeline

There are two ways to run a pipeline:

1. Run all the nodes of the pipeline sequentially, starting with the **Data** node. This is done by clicking the Run Pipeline button in the upper right corner of the canvas. This can also be done by clicking the ⋮ icon next to the current pipeline tab in the upper left corner of the canvas and clicking **Run**.
2. Run one branch of the pipeline, running only the selected node, and all nodes preceding that node by arrows. This is done by right-clicking a node, and selecting **Run**. For the pipeline to have been fully considered as having run in SAS Visual Data Mining and Machine Learning and SAS Visual Forecasting, you must use the **Model Comparison** node to run all the nodes in the pipeline.

To interrupt a running pipeline, click the ⋮ icon next to the current pipeline tab in the upper left corner of the canvas and click **Stop**.

Available Templates

The following Node templates are included with Model Studio:

Node Name	Node Description	Product
Anomaly Detection	Identifies and excludes anomalies (observations) using the support vector data description.	Data Mining and Machine Learning
Auto-forecasting	Use an ESM, ARIMAX, IDM, or UCM model to generate forecasts.	Forecasting
Batch Code	Runs SAS batch code.	Data Mining and Machine Learning
Bayesian Network	Fits a Bayesian network model for a class target.	Data Mining and Machine Learning
Categories	Classifies documents by subject.	Text Analytics
Clustering	Performs observation-based clustering for segmenting data.	Data Mining and Machine Learning
Concepts	Extracts specific information from text.	Text Analytics

Node Name	**Node Description**	**Product**
Data Exploration	Displays summary statistics and plots for variables in your data table.	Data Mining and Machine Learning
Decision Tree	Fits a classification tree for a class target or a regression tree for an interval target.	Data Mining and Machine Learning
Ensemble	Creates a new model by taking a function of posterior probabilities (for class targets) or the predicted values (for interval targets) from multiple models.	Data Mining and Machine Learning
External Forecasts	Reads forecasts that are generated by an external source.	Forecasting
Feature Extraction	Generates features based on PCA, robust PCA, SVD, or autoencoders to use as inputs. Note that PCA, SVD, and RPCA use interval inputs only.	Data Mining and Machine Learning
Filtering	Excludes observations from analysis based on specified criteria.	Data Mining and Machine Learning
Forest	Fits a forest model, which consists of multiple decision trees based on different samples of the data and different subsets of inputs.	Data Mining and Machine Learning
GLM	Fits a generalized linear model for an interval target with a specified target distribution and link function.	Data Mining and Machine Learning
Gradient Boosting	Fits a gradient boosting model, which builds a sequential series of decision trees.	Data Mining and Machine Learning
Hierarchical Forecasting	Generates forecasts for each level of the specified hierarchy.	Forecasting
Hierarchical Forecasting (Pluggable)	Generates forecasts using hierarchical forecasting model.	Forecasting
Imputation	Imputes missing values for class and interval inputs using the specified methods.	Data Mining and Machine Learning
Linear Regression	Fits an ordinary least squares regression model for an interval target.	Data Mining and Machine Learning
Logistic Regression	Fits a logistic regression model for a binary or nominal target.	Data Mining and Machine Learning
Manage Variables	Modifies the metadata of variables.	Data Mining and Machine Learning
Multistage Model	Generates forecasts using a multistage forecasting model.	Forecasting
Naive Model	Generates forecasts using naive model.	Forecasting
Neural Network	Fits a fully connected neural network model.	Data Mining and Machine Learning

Node Name	Node Description	Product
Non-seasonal Model	Generates forecasts using a non-seasonal ESM, ARIMAX, or UCM model.	Forecasting
Open Source Code	Runs Python or R code.	Data Mining and Machine Learning
Panel Series Neural Network	Generates forecast using fully connected neural network model.	Forecasting
Quantile Regression	Fits a quantile regression model for an interval target.	Data Mining and Machine Learning
Replacement	Replaces data values such as outliers and unknown class levels with specified values.	Data Mining and Machine Learning
Retired Series	Generates forecasts for retired series using a specified value.	Forecasting
SAS Code	Runs SAS code.	Data Mining and Machine Learning
Save Data	Saves data exported by a node in a pipeline to a CAS library.	Data Mining and Machine Learning
Score Code Import	Imports SAS score code.	Data Mining and Machine Learning
Score Data	Scores a table using the score code generated by predecessor nodes and saves the scored table to a CAS library.	Data Mining and Machine Learning
Seasonal Model	Generates forecasts using a seasonal ESM, ARIMAX, or UCM model.	Forecasting
Segment Profile	Examines segmented data and enables identification of factors that differentiate the segments from the population.	Data Mining and Machine Learning
Sentiment	Analyzes attitudes expressed in documents.	Text Analytics
Stacked Model (NN + TS) Forecasting	Generates forecasts using stacked model (Neural Network + Time Series).	Forecasting
SVM	Fits a support vector machine via interior-point optimization for a binary target.	Data Mining and Machine Learning
Temporal Aggregation Model	Generates forecasts using a temporal aggregation model.	Forecasting
Text Mining	Parses and performs topic discovery to prepare text data for modeling.	Data Mining and Machine Learning
Text Parsing	Prepares text for terms analysis.	Text Analytics
Time Series Regression	Generates forecasts using a regression model.	Forecasting
Topics	Assigns documents to topics.	Text Analytics

Node Name	Node Description	Product
Transformations	Applies numerical or binning transformations to input variables.	Data Mining and Machine Learning
Variable Clustering	Performs variable clustering to reduce the number of inputs.	Data Mining and Machine Learning
Variable Selection	Performs unsupervised and several supervised methods of variable selection to reduce the number of inputs.	Data Mining and Machine Learning

The following Pipeline templates are included with Model Studio:

Pipeline Name	Pipeline Description	Product
Advanced template for class target	Extends the intermediate template for class target with neural network, forest, and gradient boosting models, as well as an ensemble.	Data Mining and Machine Learning
Advanced template for class target with autotuning	Advanced template for class target with autotuned tree, forest, neural network, and gradient boosting models.	Data Mining and Machine Learning
Advanced template for interval target	Extends the intermediate template for interval target with neural network, forest, and gradient boosting models, as well as an ensemble.	Data Mining and Machine Learning
Advanced template for interval target with autotuning	Advanced template for interval target with autotuned tree, forest, neural network, and gradient boosting models.	Data Mining and Machine Learning
Auto-forecasting	Forecasting pipeline with automatic modeling.	Forecasting
Auto-forecasting (Intermittent)	Forecasting pipeline with automatic, intermittent modeling.	Forecasting
Base Forecasting	Forecasting pipeline with no modeling components added by default.	Forecasting
Basic template for class target	A simple linear flow: Data, Imputation, Logistic Regression, Model Comparison.	Data Mining and Machine Learning
Basic template for interval target	A simple linear flow: Data, Imputation, Linear Regression, Model Comparison.	Data Mining and Machine Learning
Blank Template	A Data Mining pipeline that contains only a data node.	Data Mining and Machine Learning
Demand Classification	Forecasting pipeline with demand classification segmentation.	Forecasting
External Forecasts	Forecasting pipeline with external forecasts.	Forecasting
External Segmentation	Forecasting pipeline with external segmentation.	Forecasting

Pipeline Name	Pipeline Description	Product
Feature engineering template	Data mining pipeline that performs feature engineering.	Data Mining and Machine Learning
Hierarchical Forecasting	Forecasting pipeline with hierarchical modeling.	Forecasting
Intermediate template for class target	Extends the basic template with a stepwise logistic regression model and a decision tree.	Data Mining and Machine Learning
Intermediate template for interval target	Extends the basic template with a stepwise linear regression model and a decision tree.	Data Mining and Machine Learning
Naive (Moving Average) Forecasting	Forecasting pipeline with naive, moving average modeling.	Forecasting
Naive Forecasting	Forecasting pipeline with naive modeling.	Forecasting
Non-seasonal Forecasting	Forecasting pipeline with non-seasonal modeling.	Forecasting
Regression Forecasting	Forecasting pipeline with regression modeling.	Forecasting
Retired Forecasting	Forecasting pipeline with retired modeling.	Forecasting
Seasonal Forecasting	Forecasting pipeline with seasonal modeling.	Forecasting
Text Analytics: Assisted Concept Rule Creation	Use Textual Elements to quickly generate custom concept rules.	Text Analytics
Text Analytics: Data Access	Text Analytics pipeline that contains a single Data node.	Text Analytics
Text Analytics: Generate Concepts, Topics, and Categories	Text Analytics pipeline for model generation with Concepts, Text Parsing, Sentiment, Topics, Categories.	Text Analytics
Text Analytics: Topic Discovery	Text Analytics pipeline that uses text parsing and machine learning to discover topics.	Text Analytics

Creating a New Template in The Exchange

1 Click the ⇆ icon in the upper left corner of the screen. The Exchange page opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates for SAS Visual Data Mining and Machine Learning, SAS Visual Text Analytics, and SAS Visual Forecasting applications.

2 To create a new template, select the existing template most similar to your desired template. You will duplicate and modify this template.

3 Click the ⋮ icon in the upper right corner of the screen and select **Duplicate**

4 The Save Node to The Exchange window appears. Enter a name and a description for the new template.

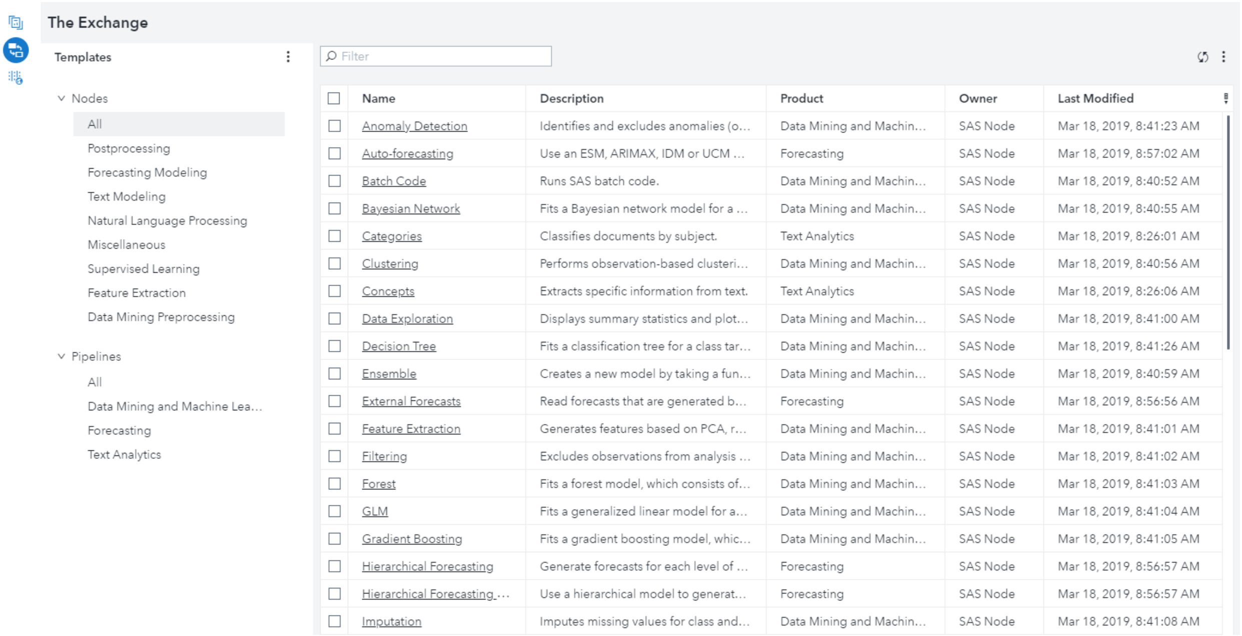
5 Click **Save**. Your new template appears in the list of templates.

Modifying an Existing Template

If you have sufficient permissions, you can modify existing templates. To modify a template:

1 Click the icon in the upper left corner of the screen.

2 The Exchange page opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates for SAS Visual Data Mining and Machine Learning, SAS Visual Text Analytics, and SAS Visual Forecasting applications.



□	Name	Description	Product	Owner	Last Modified
□	Anomaly Detection	Identifies and excludes anomalies (o...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:23 AM
□	Auto-forecasting	Use an ESM, ARIMAX, IDM or UCM ...	Forecasting	SAS Node	Mar 18, 2019, 8:57:02 AM
□	Batch Code	Runs SAS batch code.	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:40:52 AM
□	Bayesian Network	Fits a Bayesian network model for a ...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:40:55 AM
□	Categories	Classifies documents by subject.	Text Analytics	SAS Node	Mar 18, 2019, 8:26:01 AM
□	Clustering	Performs observation-based clusteri...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:40:56 AM
□	Concepts	Extracts specific information from text.	Text Analytics	SAS Node	Mar 18, 2019, 8:26:06 AM
□	Data Exploration	Displays summary statistics and plot...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:00 AM
□	Decision Tree	Fits a classification tree for a class tar...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:26 AM
□	Ensemble	Creates a new model by taking a fun...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:40:59 AM
□	External Forecasts	Read forecasts that are generated b...	Forecasting	SAS Node	Mar 18, 2019, 8:56:56 AM
□	Feature Extraction	Generates features based on PCA, r...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:01 AM
□	Filtering	Excludes observations from analysis ...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:02 AM
□	Forest	Fits a forest model, which consists of...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:03 AM
□	GLM	Fits a generalized linear model for a...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:04 AM
□	Gradient Boosting	Fits a gradient boosting model, whic...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:05 AM
□	Hierarchical Forecasting	Generate forecasts for each level of ...	Forecasting	SAS Node	Mar 18, 2019, 8:56:57 AM
□	Hierarchical Forecasting ...	Use a hierarchical model to generat...	Forecasting	SAS Node	Mar 18, 2019, 8:56:57 AM
□	Imputation	Imputes missing values for class and...	Data Mining and Machin...	SAS Node	Mar 18, 2019, 8:41:08 AM

3 To access a particular template, click the template name. This opens the Node Template or Pipeline Template window. If you do not have Edit privileges for a given template, you will see **(Read-Only)** displayed in the window.

In the Node Template or Pipeline Template window, you can make changes and configure the nodes in the pipeline. Changes are saved automatically to the template.

Note: While you are editing a template, nodes can be re-configured, but no nodes can be added or deleted.

Registering Models

After a pipeline has successfully run, you can register a Concepts, Sentiment, Topics, or Categories model. In order to register a model, complete the following steps:

1 Run the pipeline containing the node for the type of model that you want to register.

2 When the pipeline has been run successfully, right-click the node in the **Pipelines** tab for the model that you want to register and select **Register model**. When the model has been registered, a message confirms that the model was registered successfully.

When a model is registered, it appears in your model repository, which is accessible through SAS Environment Manager. The standard software deployment includes the Model Repository service. To access the model repository, complete the following steps:

1. In the upper left corner of the **Pipelines** tab, click ≡ and select **Manage Environment**.
2. On the left side of the SAS Environment Manager page, click 🗎 to access the Content window.
3. In the upper left corner of the Content window is a list of folders under **SAS Content**. One of those folders is labeled **Model Repositories**, which contains the **VTARepository** folder where any Text Analytics models that you register are stored.

Note: If you are in a CASHostAccountRequired custom group, you must follow the instructions under "File System Directory Permissions" in *SAS Viya Administration: Models* in order to register models.

SAS Environment Manager enables you to manage access for any models present in the common model repository. For more information, see "Access to Models" in *SAS Viya Administration: Models*. If you have a SAS Model Manager license, any models that you register appear in the SAS Model Manager Projects and Models category views. If you obtain a license for SAS Model Manager after you have already registered a model, that model automatically appears in the Projects and Models category views of SAS Model Manager.

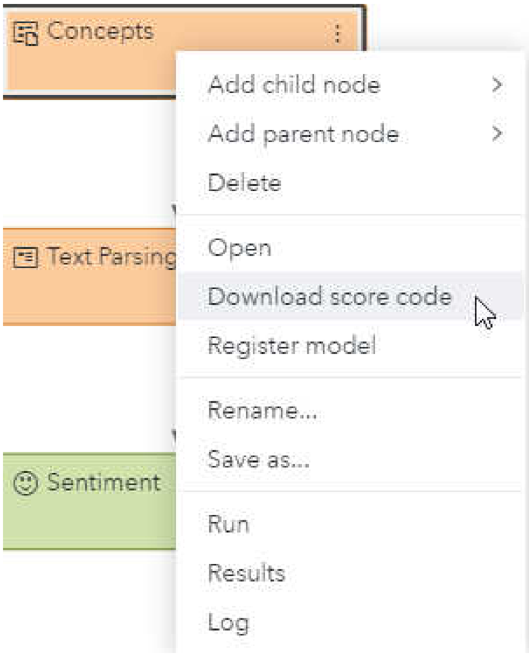
Scoring an External Data Set

You can export score code and the models that you create in a SAS Visual Text Analytics project, and apply them to a holdout data set. Score code can be viewed and downloaded from the following nodes:

- Concepts
- Sentiment
- Topics
- Categories

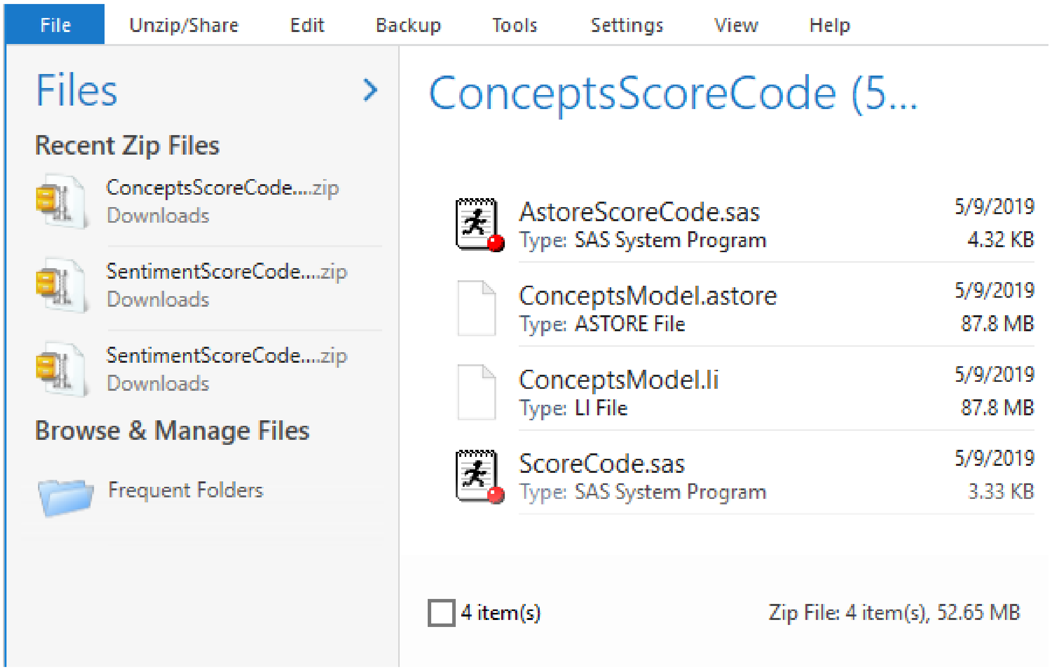
In order to download score code from an analysis node, complete the following steps:

1. Navigate to the **Pipelines** tab, and run the pipeline.
2. When the pipeline run is complete, right-click on the analysis node that you want to download the score code for and select **Download score code**.



This creates a ZIP file. The following describes what the contents of the ZIP file are depending on the node type from which you download score code. This code can be used to score an external CAS table within a SAS Viya environment (for example, in SAS Studio).

- When you download score code from a **Concepts** node, the resulting ZIP file contains the following: the concepts model (**ConceptsModel.li**) and its associated score code (**ScoreCode.sas**), and the concepts analytic store (**ConceptsModel.astore**) and its associated score code (**AstoreScoreCode.sas**).



AstoreScoreCode.sas Type: SAS System Program	5/9/2019	4.32 KB
ConceptsModel.astore Type: ASTORE File	5/9/2019	87.8 MB
ConceptsModel.li Type: LI File	5/9/2019	87.8 MB
ScoreCode.sas Type: SAS System Program	5/9/2019	3.33 KB

□ 4 item(s) Zip File: 4 item(s), 52.65 MB

- When you download score code from a **Sentiment** node, the contents of the resulting ZIP file can vary. When you download score code from a **Sentiment** node in a project that uses the base sentiment model, the ZIP file contains only **ScoreCode.sas**. However, if you specify a custom sentiment model, the ZIP file contains the following: the sentiment model (**SentimentModel.sam**) and its associated score code (**ScoreCode.sas**), and the sentiment analytic store (**SentimentModel.astore**) and its associated score code (**AstoreScoreCode.sas**).

- When you download score code from a **Topics** node, the resulting ZIP file contains the topics analytic store (**TopicsModel.astore**) and its associated score code (**AstoreScoreCode.sas**).
- When you download score code from a **Categories** node, the resulting ZIP file contains the following: the categories model (**CategoriesModel.mco**) and its associated score code (**ScoreCode.sas**), and the categories analytic store (**CategoriesModel.astore**) and its associated score code (**AstoreScoreCode.sas**).

6

Using the Concepts Node

Overview

The **Concepts** node enables you to work with semantic attributes, entity types, facts, or relationships, and extracts pieces of the text using rules written in the language interpretation for textual information (LITI) syntax. For more information about the **Concepts** node, see the following:

- “Specifying Settings for the Concepts Node” on page 39
- “Using the Interactive Window for the Concepts Node” on page 40
- “Using the Results Window for the Concepts Node” on page 48

Specifying Settings for the Concepts Node

You can adjust settings for the **Concepts** node using the options panel in the **Pipelines** tab. When you click the **Concepts** node, the options panel appears to the right of the pipeline.

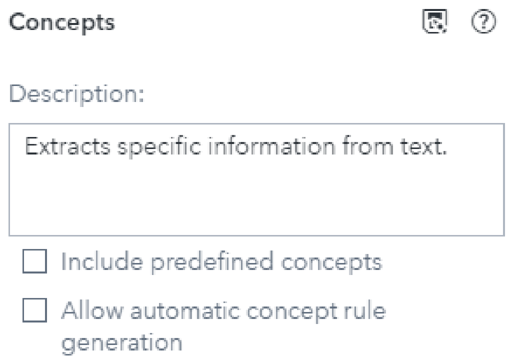
Note: You must rerun the **Concepts** node to see the results of any changes that you make to these settings.

The following options can be specified for the **Concepts** node.

- Include predefined concepts in your analysis. Predefined concepts identify items in context such as a person, location, or an organization. They save time by providing you with out-of-the-box definitions for commonly used concepts. (Predefined concept availability depends on the project data language.)
- Allow automatic concept rule generation. You can select a custom concept for automatic concept rule generation, which suggests new rules based on the existing rules for that concept. For more information

about automatically generating concept rules, see "Automatically Generate Concept Rules (Experimental)" on page 43.

In order to change these settings, select or deselect the appropriate options in the options panel for the **Concepts** node.



Concepts

Description:

Extracts specific information from text.

□ Include predefined concepts

□ Allow automatic concept rule generation

Using the Interactive Window for the Concepts Node

Using Predefined Concepts

SAS Visual Text Analytics provides *predefined concepts*, which are concepts whose rules are already written. You can provide additional rules in the rule editor of the **Edit Concept** tab to modify or extend their behavior. Predefined concepts save time by providing you with commonly used concepts and their definitions, such as an organization name or a date. You cannot rename predefined concepts, nor can you view or edit their base definitions. Predefined concepts have preset priority values that are used to determine which matches are returned when overlapping matches occur. For information about including predefined concepts in your analysis, see "Specifying Settings for the Concepts Node" on page 39. For a list of predefined concepts and their priority values for all languages, see "About Priority Values for Predefined Concepts" on page 148.

Considerations when Creating a Custom Concept

When you name a custom concept, keep the following in mind:

- Use valid characters: numbers, alphabetic letters, and underscores (_). For more information about the use of underscores and double-byte characters, see the Note at the end of this list.
- Concept names are case-sensitive.
- Create names that are not regular words. Use mixed case to help with readability. For example, MyConcept or myConcept are good names. Do not use names for custom concepts that are also words (for example, Problem or Mechanics) that could be matched in your text. Instead, use names that cannot be interpreted as words, such as MyNewConcept.

  Note: Concept names can contain only single-byte characters. Languages that have double-byte letters and characters should use only ASCII letters in names.

If underscores (_) are used in concept names, follow these guidelines to ensure that your concept rules work as expected:

- If you use underscores at either end of a concept name, there must be a matching underscore at the other end of the concept name as well. For example, `_Domestic_` is permitted, but `Domestic_` is not permitted.

- Consecutive underscores are not typically permitted in concept names. However, consecutive underscores can be used when there are matching underscores at the beginning and end of a concept name. For example, `_Country__Names_` is permitted, but `Country__Names` is not permitted.
- Do not include `_Q` anywhere in a concept name. This character combination is reserved by the application,
- If a concept name begins with an underscore, the next character must be a letter. For example, the concept name `_25anniv_` is not permitted.

Matching documents are shown only for concepts with the concept behavior set to **Primary**. Concepts with the concept behavior set to **Supporting** will not yield any matching documents. In order to change the concept behavior from **Primary** to **Supporting** for a custom concept, right-click the custom concept and select **Set concept behavior** ⇨ **Supporting**.

Note: When custom concepts are present in the **Concepts** node, the concept behavior setting of at least one custom concept should be set to **Primary**.

Creating Custom Concepts

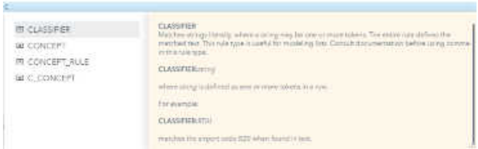
Custom concepts are user-created concepts that are defined by a set of rules that you specify. The following directions assume that you are already in the interactive window for the **Concepts** node. Before you create a custom concept, familiarize yourself with the guidelines provided in “Considerations when Creating a Custom Concept” on page 40. In order to create a custom concept, complete the following steps:

1 Select **Custom Concepts** in the **Concepts** panel, and click ☐. The Add Custom Concept window appears.

2 Enter the concept name in the Add Custom Concept window, and click **OK**.

Note: When you create a custom concept, a **Sandbox** tab is created alongside the **Edit Concept** tab. For more information about using a sandbox environment, see “Using the Sandbox Tab” on page 42.

3 Using LITI syntax, create the rules for your custom concept in the rule editor of the **Edit Concept** tab. When writing rules for concept extraction, the autocomplete feature enables users to view and select rule types based on text entered by the user.



CLASSIFIER

[illegible]

CLASSIFIER:string

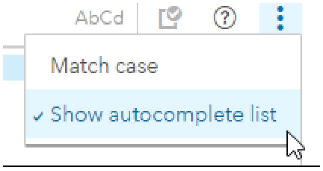
[illegible]

For example:

CLASSIFIER:[illegible]

[illegible]

If you want to disable this feature, deselect the **Show autocomplete list** option from the drop-down menu in the **Edit Concept** toolbar.



For more information about using LITI syntax, see “Writing Concept Rules: Basic LITI Syntax” on page 83.

> **TIP** Save time by using the **Sandbox** tab to test concept rules before adding them to a custom concept. Only the rules in the sandbox, and the concepts that they depend upon, are compiled into the model when testing in a sandbox environment. This enables fast and thorough testing of experimental rules against your document collection. For more information about using a sandbox, see “Using the Sandbox Tab” on page 42.

4 When you finish creating rules for your custom concept, click ☑ in the toolbar in the **Edit Concept** pane to validate the rules. If any syntax errors are identified, they must be fixed before you can run the **Concepts** node.

5 Once the rules for your custom concept have been validated, click **Run Node** in the upper right corner of the page. This ensures that only documents matching the most recent criteria will show in the matched documents tab.

Note: If you duplicate a concept, you must rerun the **Concepts** node.

Using the Sandbox Tab

In addition to the **Edit Concept** tab, each custom concept and predefined concept has a **Sandbox** tab associated with it. Unless a concept is deleted, the associated sandbox remains paired with that concept. The following features are offered in the **Sandbox** tab:

- Use the sandbox to test new rules so that you only see the results for those rules.
- Test subsets of your model against a document collection. Only the rules in the sandbox, and the concepts that they depend on, are compiled into the model when testing in a sandbox environment. This enables faster testing of your model against your document collection.
- Store any rules that are not yet ready for production in the sandbox, along with any documentation about your concept.
- Easily add new rules from the sandbox into the associated concept once testing of those rules results in the expected behavior.

When using the **Sandbox** tab, consider the following:

- If you choose to reference the rules of the concept associated with the sandbox, be careful not to move those rules back to the main concept. This action results in a circular reference error.
- The REMOVE_ITEM rule cannot be used to filter matches from a sandbox, as the sandbox itself cannot be referenced in other rules.
- Only one sandbox at a time can display document matches. Subsequent sandbox runs replace previous sandbox results.

Create Concept Rules from Terms in the Document Collection

CLASSIFIER rules can be created simply and swiftly by using the **Add rule to concept** feature in the **Documents** tab. In order to create CLASSIFIER rules from text in the **Documents** tab, complete the following steps:

1 Select either a predefined or custom concept in the **Concepts** panel.

2 Using your cursor, highlight the text in the **Documents** tab that you want to add as a classifier.

3 Click + in the upper right hand corner of the **Documents** tab. A new CLASSIFIER rule is created for the selected concept.

4 Click ☑ in the upper right hand corner of the **Edit Concept** tab to validate the concept rules, and rerun the **Concepts** node.

Automatically Generate Concept Rules (Experimental)

When you run a **Concepts** node that contains a custom concept, you can select that concept for automatic generation of concept rules. This feature creates and suggests concept rules that you might want to add to your concept, and it does so based on the ambiguity and frequency of each concept rule. Rules that appear more frequently have a higher rating, and are therefore deemed more useful than less frequently occurring rules. If you run automatic concept rule generation for multiple concepts simultaneously, this feature ensures that the same rule does not get generated for more than one concept. Using this feature enables you to optimize the effectiveness of each custom concept that you create, and greatly reduce the amount of time that you spend creating rules for custom concepts.

Note: This is an experimental feature. The algorithm that is currently used to automatically generate concept rules might change in the future.

Circular dependencies in rules can cause your model to fail, or otherwise run incorrectly. In order to avoid circular dependencies when using automatic concept rule generation, keep the following in mind:

- If you want to add rules to a predefined concept, create a custom concept that references the predefined concept, and then add your rules to that custom concept. For example, if you want to add rules to the predefined concept **`nlpPlace`**, create a custom concept with the rule `CONCEPT:nlpPlace`, and then append any other rules that you want to add. This ensures that rule generation will not generate any circular dependencies.
- Avoid using concept names that are normal tokens in your data.

The types of rules that can be created are as follows: `CONCEPT_RULE`, `C_CONCEPT`, and `CLASSIFIER`. The following describes use case scenarios for each of these rule types.

- CLASSIFIER

  Generated CLASSIFIER rules are especially useful when they use C_CONCEPT rules as input. They enable you to see what the target content entails as represented by these rules. For example, suppose you are working with a medical data set, and you want to find body parts found on the left side of the body. To accomplish this task, you might use an original rule such as `C_CONCEPT:left_c{_w}`.

- C_CONCEPT

  These rules help you establish the context around matching items. The elements that are not inside the `_c{}` modifier are the contextual elements.

  Concept rule generation for the C_CONCEPT rule type uses n-gram templates to create rules, which are generated based on matches for the original rules as well as the context of those matches. When generating rules based on context, up to two tokens on either side of an existing match are used to identify contextual patterns. The matches of these rules are then ranked, and a subset of the rules are returned to the user.

- CONCEPT_RULE

  These rules appear when you include predefined concepts in a **Concepts** node. These rules help you home in on the context in which matches are found by requiring the presence of additional elements in the context of a sentence. These contextual elements can be literal strings that represent noun groups or references to certain predefined concepts, including nlpTime, nlpDate, nlpMoney, and nlpPercent.

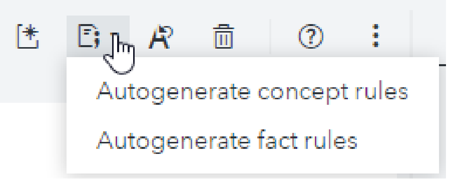
Newly-generated rules are appended to the bottom of the sandbox. Timestamp comments are located above and below the rules.

Automatically generated concept rules are generally expected to have low precision, and potentially higher recall. CLASSIFIER and C_CONCEPT rules that are generated expand the scope of the matches returned for the original set of rules. In contrast, rules of the CONCEPT_RULE type suggest methods for narrowing the original rules. This feature is particularly useful for identifying good data for modeling. If no rules are generated, check the log for the **Concepts** node to review messages.

Note: The concept that is marked for automatic concept rule generation must have matches in order to generate new rules. As a result, rules cannot be generated for concepts whose concept behavior is set to `Supporting`. There is no limit to the number of matches that a concept can have.

In order to automatically generate concept rules, complete the following steps:

1. Navigate to the **Pipelines** tab, and select the **Concepts** node.
2. Select **Allow automatic concept rule generation** in the options panel for the **Concepts** node.
3. Right-click on the **Concepts** node and select **Open**.
4. Select a custom concept from the **Concepts** panel. If you have not created a custom concept, follow the steps in Creating Custom Concepts on page 41 in order to do so.
5. In the toolbar above the **Concepts** panel, click ▤.
6. Select **Autogenerate concept rules** from the drop-down list, and an indicator appears next to the selected concept. In addition, a message appears above the rule code editor to confirm that the concept is ready for automatic rule generation.



7. Rerun the **Concepts** node in order to generate rules for the selected concept. Once the **Concepts** node is run, any rules that are generated are placed in **Sandbox** tab. No more than twenty five rules are created.

   Note: To generate more rules in addition to the original twenty-five, move the original rules from the sandbox environment to the rule editor in the **Edit Concept** tab, and repeat steps one through seven.

8. Click ⧉ in the upper right hand corner of the **Sandbox** tab. This adds all of the rules that were generated in the **Sandbox** tab to the existing concept rules.
9. Click ☑ to validate the concept rules, and rerun the concepts node.
10. If you do not want to add all of the generated rules to your concept, complete the following steps:
    a. Select the rules that you want to add to the existing concept rules by highlighting them with your cursor.
    b. Right-click inside of the rule editor in the **Sandbox** tab, and select **Copy** from the pop-up menu.
    c. Navigate to the **Edit Concept** tab, and press Ctrl+V to append the selected rules to the existing concept rules.
    d. Click ☑ to validate the concept rules, and rerun the concepts node.

Automatically Generate Fact Rules

In SAS Visual Text Analytics, there are two types of fact rules: PREDICATE_RULE and SEQUENCE. These rules are used to locate and match custom concepts that are related.

Automatic fact rule generation requires that there are at least three custom concepts present, and that at least two of those custom concepts have matches. This is because fact rule generation uses the matches found from two custom concepts that you specify to create fact rules.

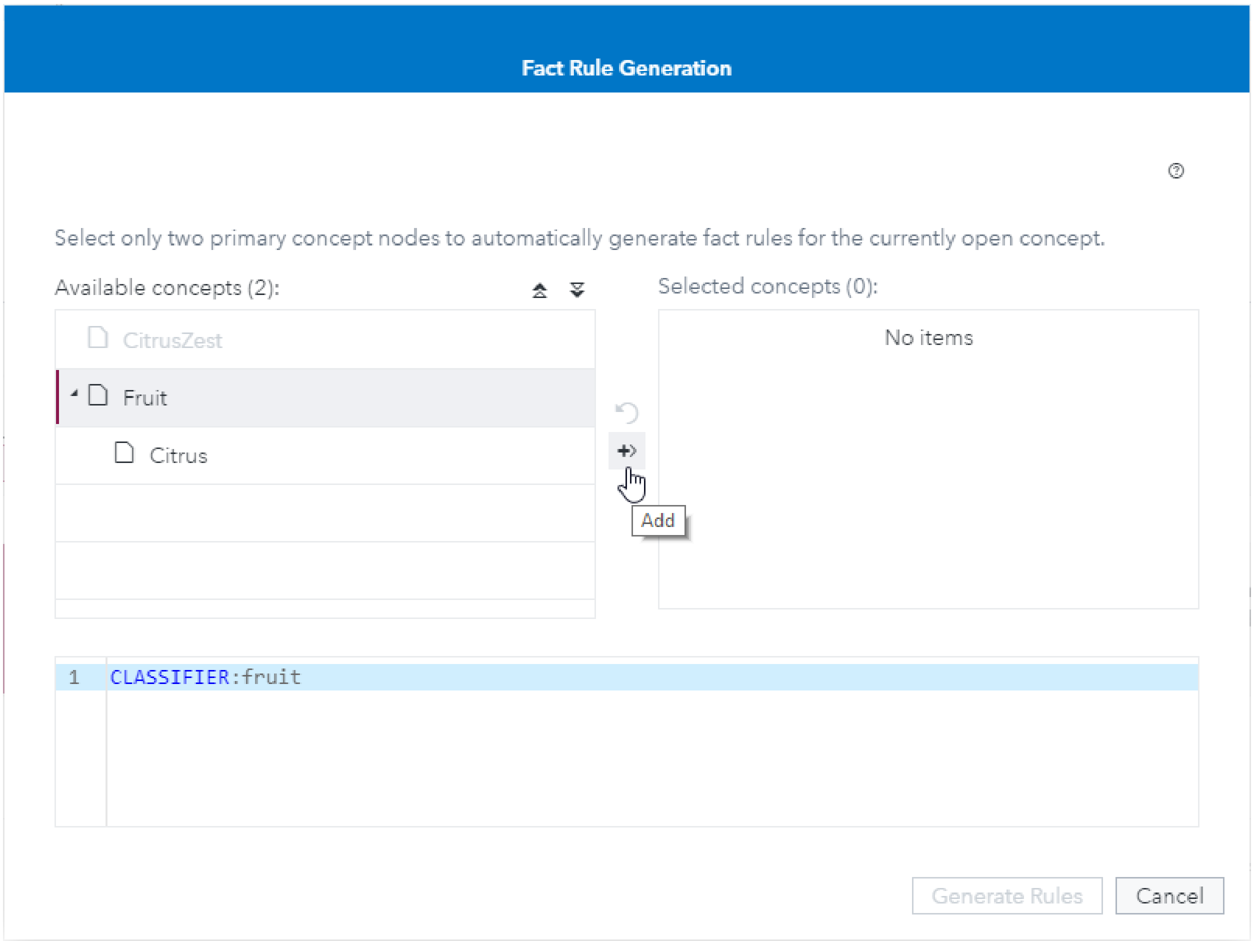
One or two rules will be generated when you use this feature. If only one rule is generated, it is a baseline rule. Baseline rules identify all matches within one sentence, and contain only the SENT operator. This rule can be extended in scope by replacing the SENT operator with one of the following operators: ORD, PARA, SENT_n, or AND.

A restricted rule can also be generated in addition to a baseline rule. These rules place restrictions on either distance, order, or both, and they use the following operators: ORDDIST_n, ORD, and DIST_n. Restricted rules identify what the most common pattern in your data looks like, and uses 85% of matches as a cutoff point to decide the following:

- Are matches usually in a particular order?
- Are matches usually within six tokens of each other?

In order to automatically generate fact rules, complete the following steps:

1. Select a custom concept that you want to generate fact rules for.
2. In the toolbar above the **Concepts** panel, click 🗎.
3. Select **Autogenerate fact rules** from the drop-down list. The Fact Rule Generation window appears.
4. Select a concept from the **Available concepts** list, and click +›.



Fact Rule Generation

Select only two primary concept nodes to automatically generate fact rules for the currently open concept.

Available concepts (2):

CitrusZest

Fruit

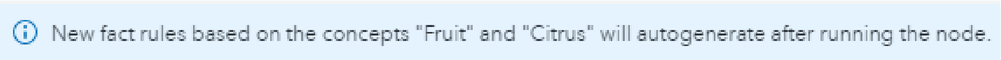
Citrus

Selected concepts (0):

No items

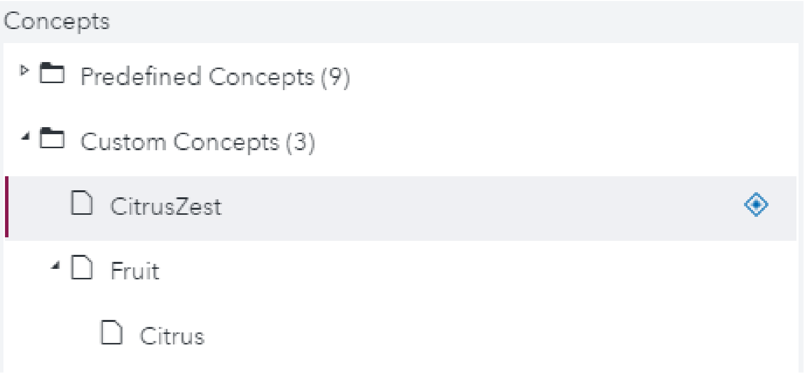
```
CLASSIFIER:fruit
```

5. Select another concept from the **Available concepts** list, and click +›.
6. Click **Generate Rules** in the lower right corner of the Fact Rule Generation window. The following message appears above the rule editor of the **Edit Concept** tab.



New fact rules based on the concepts "Fruit" and "Citrus" will autogenerate after running the node.

7 Click **Run Node** in the upper right corner of the page. Once the node runs successfully, an indicator appears next to the concept that was selected for automatic fact rule generation.



8 Navigate to the **Sandbox** tab to see the fact rules that were generated for the selected concept.

9 Click ⧉ in the upper right corner of the **Sandbox** tab. This adds all of the rules that were generated in the **Sandbox** tab to the **Edit Concept** rule editor.

10 Click ☑ to validate the concept rules, and rerun the concepts node.

11 If you do not want to add all of the generated rules to your concept, complete the following steps:

a Select the rules that you want to add to your concept by highlighting them with your cursor.

b Right-click inside of the rule editor in the **Sandbox** tab, and select **Copy** from the pop-up menu.

c Navigate to the **Edit Concept** tab, and press Ctrl+V to add the selected fact rules to your concept.

d Click ☑ to validate the concept rules, and rerun the concepts node.

For more information about facts and their components, see “Concepts versus Facts” on page 84.

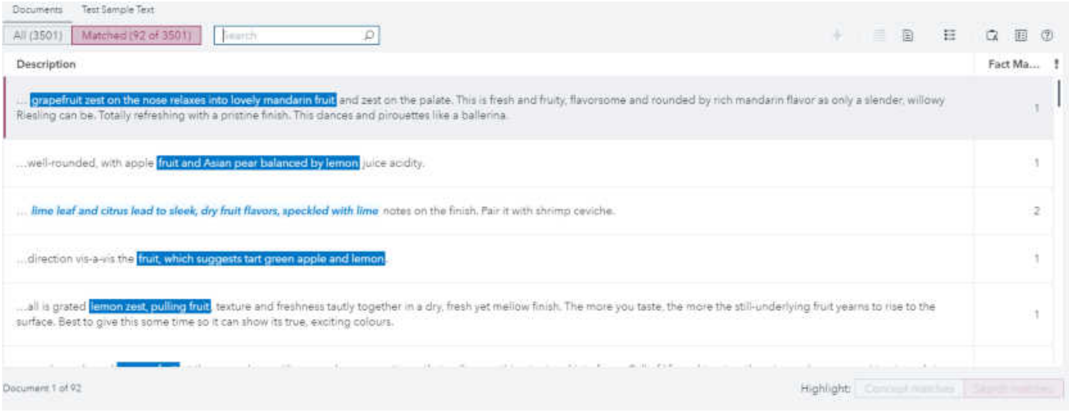
View Matching Documents by Concept

When testing a document set against a concept, three types of matches can occur. A document can contain a **Matched item**, **Matched fact**, or an **Overlapping match**. Matches are formatted uniquely based on which of the three match types they are considered. In order to test a document set against a concept, complete the following steps:

1 Select a predefined concept or a custom concept from the **Concepts** pane.

2 In the upper left corner of the **Documents** tab, select **Matched**.

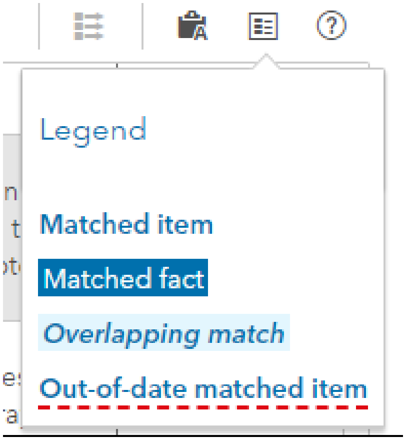
3 Each document containing a match is displayed, and each match is highlighted within the document.



Description	Fact Ma...
... grapefruit zest on the nose relaxes into lovely mandarin fruit and zest on the palate. This is fresh and fruity, flavorsome and rounded by rich mandarin flavor as only a slender, willowy Riesling can be. Totally refreshing with a pristine finish. This dances and pirouettes like a ballerina.	1
...well-rounded, with apple fruit and Asian pear balanced by lemon juice acidity.	1
... *lime leaf and citrus lead to sleek, dry fruit flavors, speckled with lime* notes on the finish. Pair it with shrimp ceviche.	2
...direction vis-a-vis the fruit, which suggests tart green apple and lemon.	1
...all is grated lemon zest, pulling fruit, texture and freshness tautly together in a dry, fresh yet mellow finish. The more you taste, the more the still-underlying fruit yearns to rise to the surface. Best to give this some time so it can show its true, exciting colours.	1

Document 1 of 92

Note: To determine the match type of each match that is discovered, select the ▤ icon in the toolbar of the **Matched** tab to access the legend.



Legend

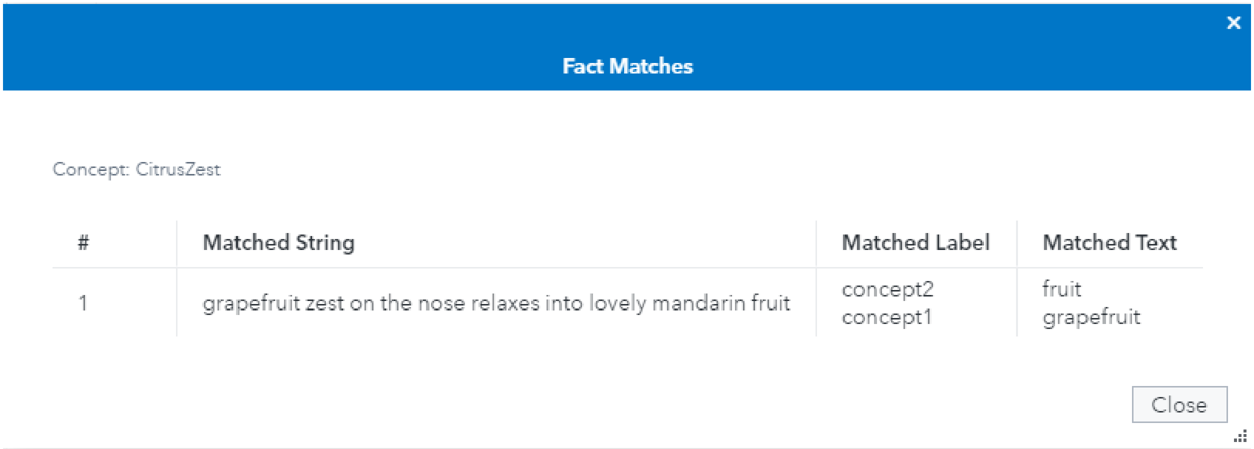
Matched item

Matched fact

Overlapping match

Out-of-date matched item

For documents that contain a matched fact, you can select that document and click the ☰ icon to view the results in greater detail. A single fact corresponds to either a `PREDICATE_RULE` or a `SEQUENCE` rule, and each rule type can have multiple matching labels. The matched string, matched label, and matched text that are associated with each matched fact are shown.



Fact Matches

Concept: CitrusZest

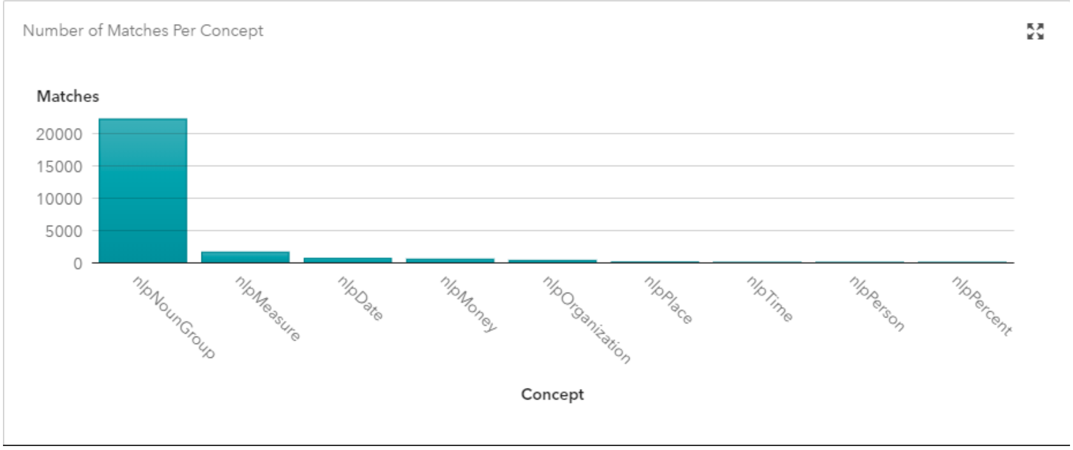
#	Matched String	Matched Label	Matched Text
1	grapefruit zest on the nose relaxes into lovely mandarin fruit	concept2 concept1	fruit grapefruit

For more information about facts and their components, see “Concepts versus Facts” on page 84.

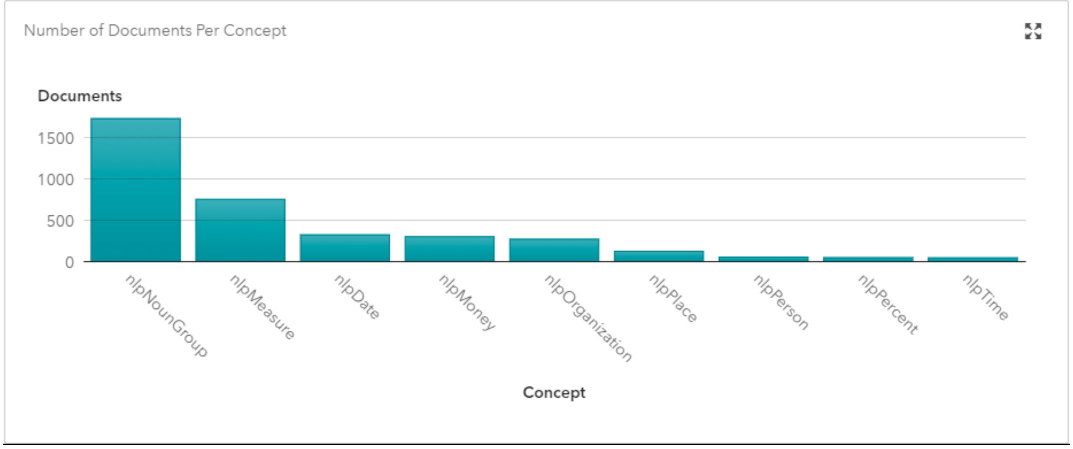
Using the Results Window for the Concepts Node

When predefined or custom concepts are included in a model, the Concepts Results window contains three bar charts, as well as the Concepts score code. The bar charts displayed are as follows:

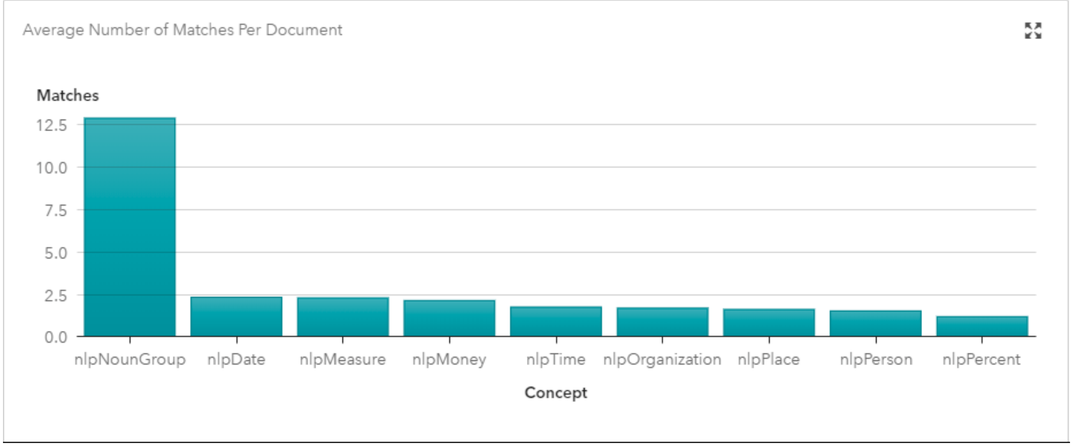
- Number of Matches Per Concept



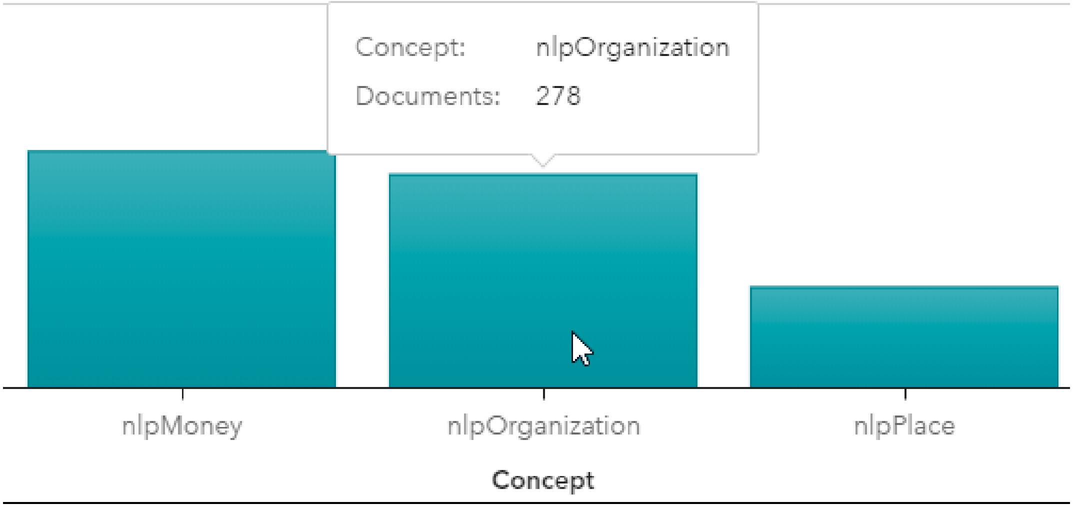
- Number of Documents Per Concept



- Average Number of Matches Per Document



If no concepts are present, only the Concepts score code is shown. Click the ↙↗ icon in the upper right corner of any of the three bar charts to maximize your view. When you maximize your view, you can position your cursor over each bar to see the document count or match count for each concept present.



When you are finished viewing a bar chart using the maximized view, click the ↘↖ icon in the upper right corner to exit. To return to the pipeline view, click the **Close** button in the upper right corner of the Concepts Results window.

7

Using the Text Parsing Node

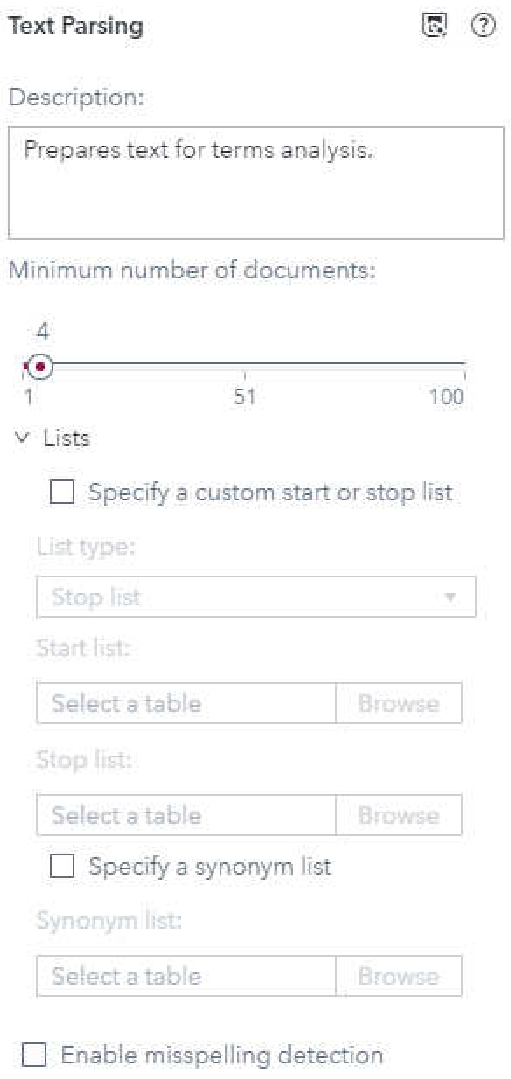
Overview

The **Text Parsing** node enables you to view and explore the terms that are present in your document collection. During the parsing process, terms are either kept or dropped based on their importance. For example, terms that have a role of preposition or conjunction often provide minimal value, and are often dropped during text parsing. To gain a better understanding of how all of your terms are related, you can generate a term map or similarity scores for a selected term to explore its relationship with other terms in your document collection. Using these tools can help you make informed decisions, such as dropping an irrelevant kept term. These changes can improve your models in downstream analysis nodes. For more information about the **Text Parsing** node, see the following:

- "Specify Settings for the Text Parsing Node" on page 51
- "Using the Interactive Window for the Text Parsing Node" on page 54
- "Using the Results Window for the Text Parsing Node" on page 57
- "Distributed Accumulation" on page 58

Specify Settings for the Text Parsing Node

You can adjust settings for the **Text Parsing** node using the options panel in the **Pipelines** tab. When you click the **Text Parsing** node, the options panel appears to the right of the pipeline.



Text Parsing

Description:

Prepares text for terms analysis.

Minimum number of documents:

4

Lists

□ Specify a custom start or stop list

List type:

Stop list

Start list:

Select a table Browse

Stop list:

Select a table Browse

□ Specify a synonym list

Synonym list:

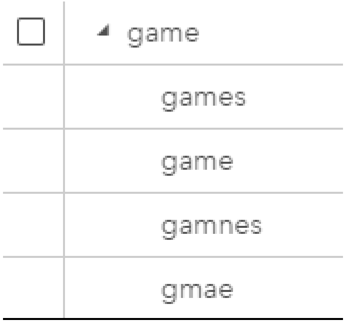
Select a table Browse

□ Enable misspelling detection

Note: You must rerun the **Text Parsing** node to see the results of any changes that you make to these settings.

The following options can be specified for the **Text Parsing** node:

- **Minimum number of documents** — this setting lets you decide the number of documents in which a term must appear in order for it to be kept during the parsing process. The default value is 4. Use the scroll bar underneath this setting to change this value.
- **Specify a custom start or stop list** — A start list specifies which terms are kept during parsing. A stop list specifies which terms should be dropped. If you do not want to use the default stop list, you can import your own start list or stop list into your project.
- **Specify a synonym list** — A synonym list is a SAS data set that identifies pairs of words that should be combined as single terms for the purposes of analysis. If you want to create custom parent terms, or group other terms under a parent term, you can specify a synonym list.
- **Enable misspelling detection** — When you enable this feature, misspelled words are identified and rolled up under the corresponding parent term. When this option is disabled, any misspelled words that are encountered are created as a separate term in the terms panes. In the image below, you will see *gmae* and *gamnes* listed as child terms of *game*.



□	◢ game
	games
	game
	gamnes
	gmae

In order to import a custom start or stop list, complete the following steps:

1 Navigate to the **Pipelines** tab and click the **Text Parsing** node.

2 Locate the options panel to the right of the pipeline, and select **Specify a custom start or stop list**.

3 Select the type of list you want to specify under **List type**. Under **List type**, the **Browse** option for the selected list type becomes available.

4 Click **Browse**, and the Choose Data window appears.

5 Select the **Import** tab in the upper left corner of the Choose Data window, and navigate to the folder that contains the list that you want to import.

6 Select the list that you want to import, and click **Open** in the bottom right corner of the window. This brings you back to the Choose Data window.

7 Click **Import Item** in the upper right corner of the Choose Data window. When the custom start or stop list is successfully imported, a confirmation message appears at the top of the window.

8 Click **OK** in the bottom right corner of the Choose Data window once the list is successfully imported.

In order to import a synonym list, complete the following steps:

1 Navigate to the **Pipelines** tab and click the **Text Parsing** node.

2 Locate the options panel to the right of the pipeline, and select **Specify a synonym list**.

3 Locate the **Synonym list** field directly under the option **Specify a synonym list**, and click **Browse**. The Choose Data window appears.

4 Select the **Import** tab in the upper left corner of the Choose Data window, and navigate to the folder that contains the synonym list that you want to import.

5 Select the list that you want to import, and click **Open** in the bottom right corner of the window. This brings you back to the Choose Data window.

6 Click **Import Item** in the upper right corner of the Choose Data window. When the synonym list is successfully imported, a confirmation message appears at the top of the window.

7 Click **OK** in the bottom right corner of the Choose Data window once the synonym list is successfully imported.

Using the Interactive Window for the Text Parsing Node

Overview

The interactive window for the **Text Parsing** node consists of a **Kept Terms** panel, a **Dropped Terms** panel, and a **Documents** panel. The following sections explain the tasks that can be performed in each of these panels.

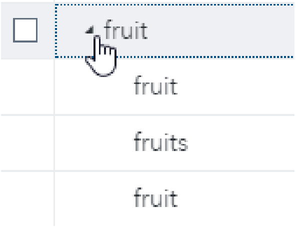
Dropping a Kept Term

Terms in the **Kept Terms** panel are terms that will generally add value to a downstream analysis node, such as the **Topics** node. However, you might choose to drop a kept term if you do not think it adds value to downstream models. For example, terms that have a low frequency of occurrence might be considered unimportant, and dropping those terms excludes them from your analysis. In order to drop a term that was kept during text parsing, complete the following steps:

1 Select a term in the **Kept Terms** panel.

2 Click ⊖ in the upper right corner of the **Kept Terms** panel. The selected term is moved to the **Dropped Terms** panel.

Note: If the selected term has any child terms, the child terms are also dropped. To see the child terms associated with a parent term, click > to the left of the parent term.



☐ ◢ fruit
fruit
fruits
fruit

3 Click **Run Node** in the upper right corner of the interactive window for the **Text Parsing** node.

Keeping a Dropped Term

Terms in the **Dropped Terms** panel are terms that are considered to provide minimal value, and are therefore excluded from analysis. However, you might choose to keep a dropped term if you think it adds value to downstream models. To keep a term that was dropped during parsing, complete the following steps:

1 Select a term in the **Dropped Terms** panel.

2 Click ⊖ in the upper right corner of the **Dropped Terms** panel. The selected term is moved to the **Kept Terms** panel.

Note: If the selected term has any child terms, the child terms are also kept.

3 Click **Run Node** in the upper right corner of the interactive window for the **Text Parsing** node.

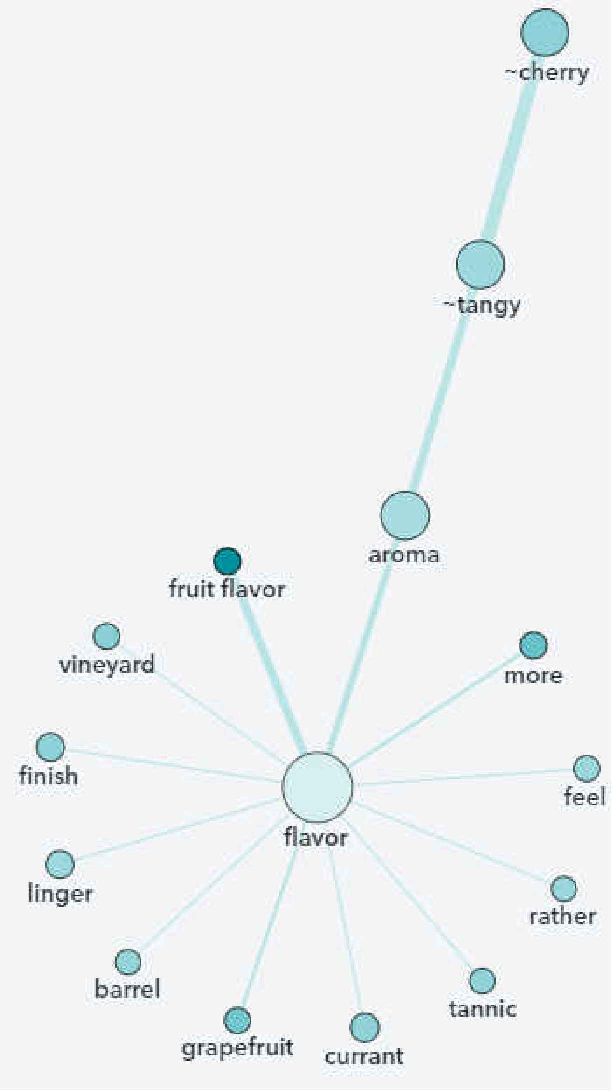
Generate Similarity Scores

Similarity scores indicate how likely it is that other terms will appear in the same context as a selected term. Similarity scores range from 0 to 1. Scores that are closer to 1 have a higher likelihood of appearing in the same context as the selected term, whereas scores closer to 0 indicate a lower likelihood. In the interactive window for the **Text Parsing** node, you can only generate similarity scores in the **Kept Terms** panel. In order to generate similarity scores for a selected term, follow the steps in "Generating Similarity Scores" on page 80.

Create a Term Map

A term map consists several nodes, where the center term node represents a selected term, and the outer term nodes represent terms that can be used to predict the presence of the selected term in a document. Consider the following information when interpreting a term map:

- The line that connects one term node to another indicates the strength of association between those two terms. A thicker line implies a stronger association between terms, and a thinner line implies a weaker association. The measure of this association is called *information gain*, which is the amount of additional information obtained by adding a conjoined term in a term map to a current rule. To see the information gain between term nodes, positiion your cursor over the line that connects the two nodes.
- The color of each term node indicates how reliably that term can be used to predict the presence of the selected term in a document. A darker term node implies greater reliability, whereas a lighter term node implies that a term is less reliable for predicting the presence of the selected term.
- While some term nodes are used to predict the presence of a selected term in a document collection, others are used to predict the absence of a selected term. Term nodes that are used to predict the absence of a selected term are preceded by a tilde(~). In the term map shown below, there is a strong relation between the term nodes **flavor** and **aroma**. There is also a strong relation between the term nodes **aroma** and **~tangy**. However, the tilde(~) in the term node **~tangy** implies that documents that contain the terms *aroma* and *tangy* are highly unlikely to contain the term *flavor*.



In order to create a term map, complete the following steps:

1 Select a term from the **Kept Terms** panel.

2 Click ✲ in the upper right corner of the **Kept Terms** panel. A term map is created for the selected term. To return to the interactive window for the **Text Parsing** node, click **Close** in the upper right corner of the page.

View Matching Documents by Term

In order to display matching documents for a term, complete the following steps:

1 In the pipeline view, right-click the **Text Parsing** node and click **Open**.

2 Select a term from the **Kept Terms** panel.

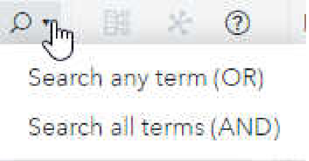
3 Click the **Matched** tab in the **Documents** panel. Matches that are found are highlighted.

Note: The **Text Parsing** node uses the “best match” method.

You can also use the **Search in documents** feature, which enables you to select and search multiple terms using either the AND or OR Boolean operator. The AND operator returns only documents that contain all of the selected terms, whereas the OR operator returns documents that contain at least one of the selected terms. If any of the selected terms have child terms, documents that match the search criteria and contain those child terms are returned as well. This feature is available in the **Kept Terms** panel, as well as the **Dropped Terms** panel. In order to search on multiple terms, complete the following steps:

1 Select at least one term from either the **Kept Terms** panel or **Dropped Terms** panel.

2 Click ⌕ in the upper right corner of the panel from which you selected terms, and select the operator that you want to use from the drop-down list.



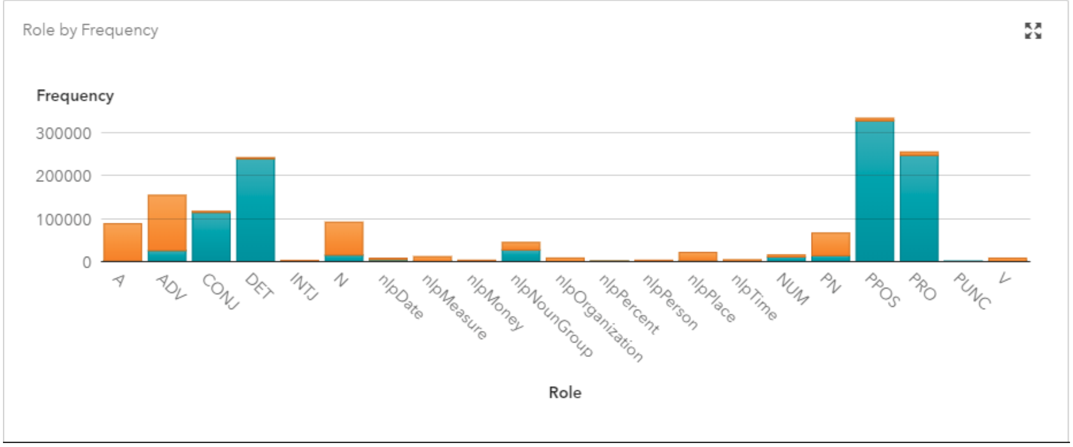
The **Documents** panel is updated to show matching documents, and each match is highlighted. Notice that when the matching documents are returned, a search query appears in the search bar of the **Documents** panel. This query is generated based on the terms and the operator that you selected. For more information about search queries in the **Documents** panel, see "Performing Searches on a Document Collection" on page 77.

Before you create another search query, clear the search query by clicking ⊗ in the search bar. Otherwise, matching documents that are returned will not reflect your new query, as any additional search queries you create are appended to the original one.

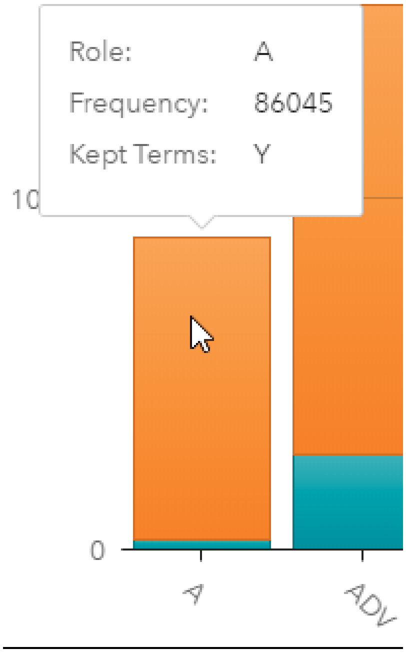
Using the Results Window for the Text Parsing Node

The Text Parsing Results window contains a bar chart titled Role by Frequency, as well as a table displaying Descriptive Statistics.

The Role by Frequency chart is a stacked bar chart, and it shows the number of times that terms of a certain role type were kept or dropped.

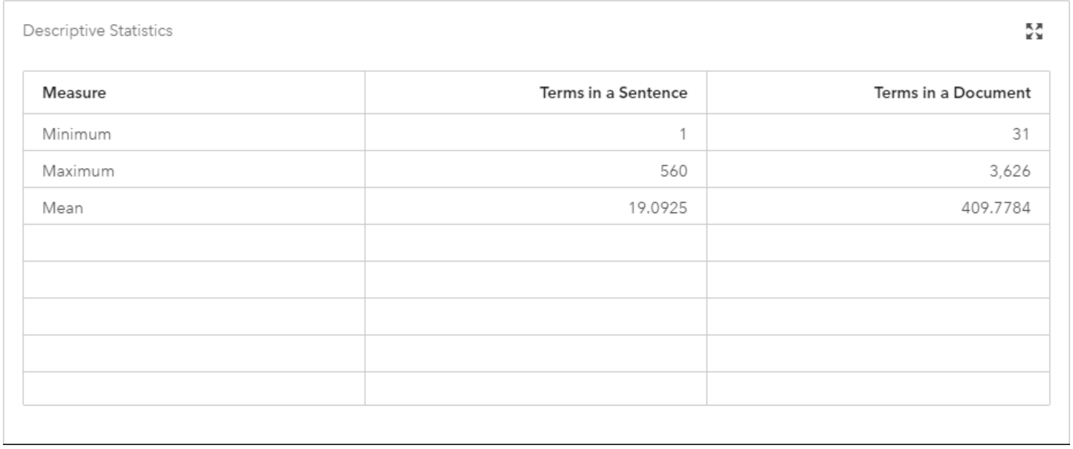


In the above image, the orange segments represent terms that were kept, and the blue segments represent terms that were dropped. Expanding the bar chart and positioning your cursor over each bar gives you a synopsis of each role.



The role type, frequency, and indication of whether the bar is representative of kept terms or dropped terms is displayed.

The Descriptive Statistics table displays the minimum, maximum, and mean for both **Terms in a Sentence** and **Terms in a Document**.



Descriptive Statistics

Measure	Terms in a Sentence	Terms in a Document
Minimum	1	31
Maximum	560	3,626
Mean	19.0925	409.7784

Distributed Accumulation

The **Text Parsing** node uses distributed accumulation for processing data. Distributed accumulation can lead to faster processing for your data by fully distributing all aspects of the accumulation process across the grid. With distributed accumulation, term counts are gathered and subtotaled at each node in the grid, and then merged into a combined total across the grid. Without distributed accumulation, term counts are gathered on a central grid node and totaled at the end of the accumulation process. Distributed accumulation also introduces extra detail to the terms table, which makes necessary information available during each step of the text analytics process.

8

Using the Sentiment Node

Overview

Sentiment analysis is the process of identifying the author’s tone or attitude (positive, negative, or neutral) expressed in a document. For more information about how sentiment scoring works, see “Sentiment Scoring” on page 9. For more information about the **Sentiment** node, see the following:

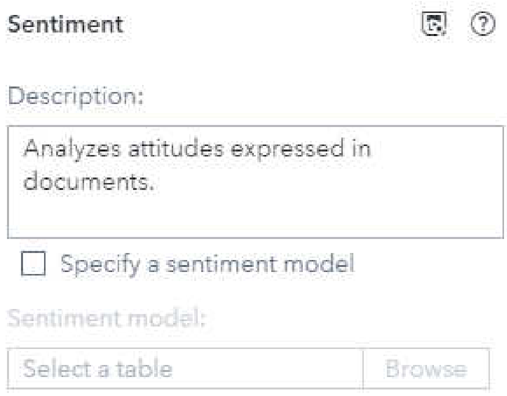
- “Specifying Settings for the Sentiment Node” on page 59
- “Using the Results Window for the Sentiment Node” on page 60

Specifying Settings for the Sentiment Node

You can adjust settings for the **Sentiment** node using the options panel in the **Pipelines** tab. When you click the **Sentiment** node, the options panel appears to the right of the pipeline. In the options panel for the **Sentiment** node, you can specify a sentiment model that you want to upload for your project. Specifying a custom sentiment model can prove especially useful if there is no base sentiment model for the selected project language. For a list of project languages that have officially supported base sentiment models, see “Sentiment Scoring” on page 9.

In order to import a sentiment model, complete the following steps:

1 In the **Pipelines** tab, click the **Sentiment** node. The options panel for the **Sentiment** node appears to the right of the pipeline.



Sentiment

Description:

Analyzes attitudes expressed in documents.

□ Specify a sentiment model

Sentiment model:

2 Select **Specify a sentiment model**, and click **Browse**. The Choose Data window appears.

3 Select the **Import** tab in the upper left corner of the Choose Data window, and navigate to the folder that contains the sentiment model that you want to use.

4 Select the sentiment model, and click **Open**.

5 In the upper right corner of the Choose Data window, click **Import Item**. A message appears at the top of the Choose Data window when the model is imported successfully.

6 Click **OK** in the bottom right corner of the Choose Data window. The **Pipelines** tab appears.

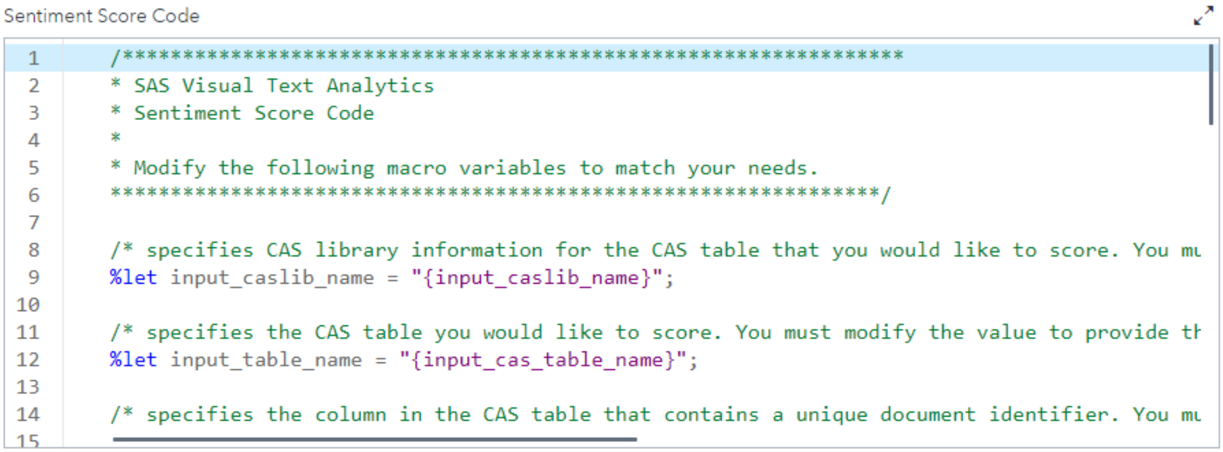
7 Right-click the **Sentiment** node, and select **Run**. The sentiment that is displayed in any nodes that are downstream of the **Sentiment** node reflects the model that you imported.

For more information about loading a Sentiment Analysis Model (SAM) file into a CAS table programmatically, see the following examples in the *SAS Visual Text Analytics: Programming Guide*:

- "Generate Sentiment Results, Match String, and Features from Input Documents" in *SAS Visual Text Analytics: Programming Guide*
- "Loading a Sentiment Binary File into a CAS Table Using the loadTableFromDisk Action" in *SAS Visual Text Analytics: Programming Guide*

Using the Results Window for the Sentiment Node

The Sentiment Results window contains the score code for the **Sentiment** node.



Sentiment Score Code

```
/******************************************************************
 * SAS Visual Text Analytics
 * Sentiment Score Code
 *
 * Modify the following macro variables to match your needs.
 *****************************************************************/

/* specifies CAS library information for the CAS table that you would like to score. You mu
%let input_caslib_name = "{input_caslib_name}";

/* specifies the CAS table you would like to score. You must modify the value to provide th
%let input_table_name = "{input_cas_table_name}";

/* specifies the column in the CAS table that contains a unique document identifier. You mu
```

For more information about using score code to score an external data set, see “Scoring an External Data Set” on page 36.

9

Using the Topics Node

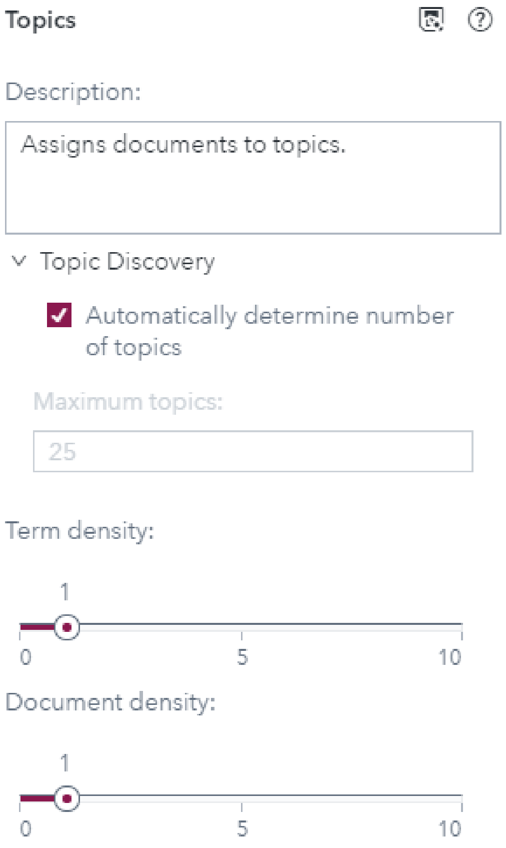
Overview

The **Topics** node enables you to find and analyze topics from your document collection. For more information about the **Topics** node, see the following:

- "Specifying Settings for the Topics Node" on page 63
- "Using the Interactive Window for the Topics Node" on page 65
- "Using the Results Window for the Topics Node" on page 67

Specifying Settings for the Topics Node

You can adjust settings for the **Topics** node using the options panel in the **Pipelines** tab. When you click the **Topics** node, the options panel appears to the right of the pipeline.



Topics

Description:

Assigns documents to topics.

Topic Discovery

☑ Automatically determine number of topics

Maximum topics: 25

Term density: 1

Document density: 1

The following options can be specified for the **Topics** node.

- The **Topic Discovery** settings determine the number of topics that are generated when you run a **Topics** node. If you want the **Topics** node to determine the number of topics that should be generated, select **Automatically determine number of topics**. If you want to specify a maximum number of topics that will be generated, deselect **Automatically determine number of topics** and enter a value in the **Maximum topics** field.
- The **Term density** setting determines the term cutoff value for each topic. For each topic in a document collection, the topic calculation computes a weight for each term indicating the influence of the term on the topic. If the absolute value of a term's weight is above the cutoff, the term is included in the topic. Terms that have absolute weights below the cutoff are not included in the topic. The term density specifies how many standard deviations above the mean of the weights to set the term cutoff.

  Specifying too low of a density for your data can result in having every single term as part of your topic. Specifying a term density that is too high for your data can result in the elimination of all terms from your topic. The typical range for term density is between 1 and 3, but if your data appears to have an abnormal distribution, you might want to use values outside of that range. Use this setting in conjunction with document density.
- The **Document density** setting affects the cutoff for each topic in a way similar to term density. Documents are assigned to a topic if the absolute value of the document weight is above the cutoff. The document density specifies how many standard deviations above the mean of the weights to set the document cutoff.

  If you want a larger number of documents to be assigned to each topic, select a lower value for document density. Increasing document density leads to fewer documents being assigned to each topic. As with term density, the typical range of values should be between 1 and 3. Use this setting in conjunction with term density.

Using the Interactive Window for the Topics Node

Overview

The interactive window for the **Topics** node includes **Topics** panel, a **Terms** panel, and a **Documents** panel. The interactive window for the **Topics** node enables you to modify and create topics that can be used to generate more effective models.

Exploring Topics in your Document Collection

Topics include groupings of important terms that are identified in a document collection. The five terms with the highest relevancy score within a topic are used to identify that topic. A relevancy score is a score that indicates how well a document satisfies a rule or model. The best match has a score of 1 and reflects a perfect (100%) match. The number of topics that are generated, and the number of terms each topic contains can vary depending on the size of the document collection. The settings that are specified in the options panel for the **Topics** node also affect the number of topics and terms. In order to see the terms that comprise each topic in your document collection, complete the following steps:

1. Select a topic in the **Topics** panel.
2. Locate the **Terms** panel to the right of the **Topics** panel, and click **Matched**. The terms that comprise the selected topic are listed by relevancy score in descending order.

Merging Topics

If two topics appear to be similar to one another, you can merge those topics into one. In order to merge two topics, complete the following steps:

1. Select the two topics that you want to merge from the **Topics** panel.
2. Click ⊞ in the upper right corner of the **Topics** panel. The modified topic appears in the **Topics** panel.
3. In the upper right corner of the interactive window for the **Topics** node, click **Run Node** to see matching documents or terms for a new topic.

Splitting Topics

If a topic seems to be too broad in scope, you can split that topic into two new topics. In order to split a topic, complete the following steps:

1. Select the topic that you want to split from the **Topics** panel.
2. Click ◫. Two new topics appear in the **Topics** panel.
3. In the upper right corner of the interactive window for the **Topics** node, click **Run Node** to see matching documents or terms for the new topics.

Create a Topic from Terms

Creating a topic from terms that you select is effective for targeting groups of documents specific to your analysis. You can also use this feature in conjunction with the merging functionality if you want to add terms to an existing topic. In order to create a topic from terms, complete the following steps:

1. Select the terms that you want to use to create a topic from the **Terms** panel.
2. Click in the upper right corner of the **Terms** panel. The new topic appears at the bottom of the **Topics** panel.
3. Click **Run Node** in the upper right corner of the interactive window for the **Topics** node to see matching documents or terms for a new topic.

Add a Topic as a Category

To add a topic as a category, complete the following steps:

1. Select the topic that you want to add as a category from the **Topics** panel.

   Note: If you add a topic as a category, and that topic name contains quotation marks, the category node will not successfully run.
2. Click in the upper right corner of the **Topics** panel.
3. Navigate to the **Pipelines** tab. Click the **Categories** node, and ensure that the option **Automatically generate categories and rules** is selected. For more information about the settings for the **Categories** node, see "Specifying Settings for the Categories Node" on page 69. Once the node is run, the two new topics appear in the **Topics** panel.
4. Right-click the **Categories** node, and select **Run**.
5. Once the **Categories** node is run, right-click the **Categories** node and select **Open**.

The topic that was added as a category appears in the **Categories** panel. To see the category rule that were generated, select the new category. The category rule is displayed in the rule editor of the **Edit Category** tab. For information about category rules, see "Writing Category Rules: Boolean Rules" on page 102.

View Matching Documents by Topic

In order to display matching documents for a topic, complete the following steps:

1. In the pipeline view, right-click the **Topics** node and click **Open**.
2. Select a topic from the **Topics** panel.
3. Click the **Matched** tab in the **Documents** panel.

Using the Results Window for the Topics Node

Overview

After a pipeline has run successfully, you can view results for the **Topics** node by right-clicking on the node and selecting **Results**. The Results window contains a **Summary** tab as well as an **Output Data** tab. These two tabs are explained in detail below. In some cases, they enable you to create output data that can be used for further modeling.

Performing Tasks in the Summary Tab

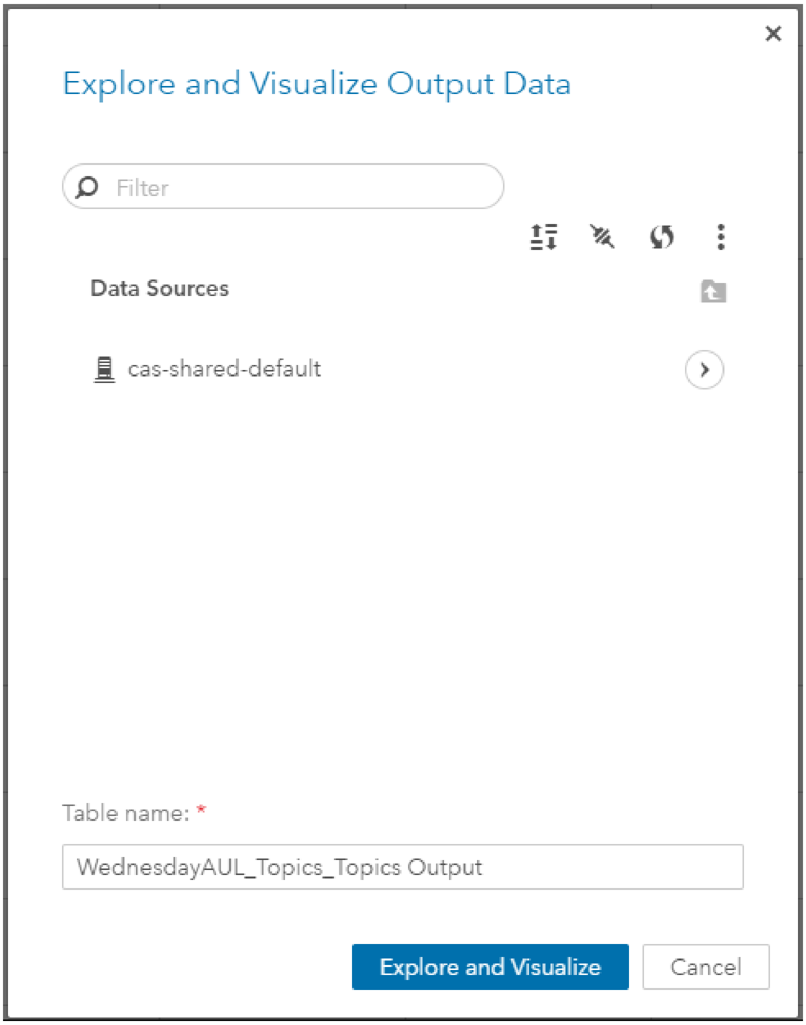
The **Summary** tab displays the bar chart **Number of Documents Per Topic**, as well as the **Topics Score Code** panel. To see the count of the number of documents per topic, expand the **Number of Documents Per Topic** bar chart by clicking ⤢ in the upper right corner of the panel. Position your cursor over each bar to display the topic name and the document count for that topic. If a **Sentiment** node precedes a **Topics** node, then the number of matching documents is displayed by sentiment within each topic. Any documents that are not assigned to a topic are accounted for in the bar labeled **No Matching Topic**.

Performing Tasks in the Output Data Tab

The **Output Data** tab is located in the upper left corner of the Topics Results window, and enables you to generate output data. A row is created for each document in the collection, and two columns are created for each topic. One column displays the score of each document for a given topic, which is expressed as a decimal. The other column displays a `0` or a `1` for each document, which indicates whether a document belongs to a given topic. In order to generate output data, open the **Output Data** tab, and click **View Output Data** in the middle of the screen. When the output data is successfully generated, the table containing that data automatically appears. You can save the output data for later use by clicking the 💾 icon in the upper left corner of the **Output Data** tab.

Note: Table names cannot exceed 247 bytes.

You can also visualize your Topics output data by clicking the 📊 icon in the upper left corner of the **Output Data** tab. The Explore and Visualize Output Data window appears, and you are prompted to select a CAS library that you want to save your output table in.



Explore and Visualize Output Data

Table name: *

WednesdayAUL_Topics_Topics Output

When you have selected a CAS library, click **Explore and Visualize** in the lower right corner of the window. This redirects you to SAS Visual Analytics, where you can use a variety of tools to model your data. For information about using SAS Visual Analytics, see SAS Visual Analytics: Getting Started with Reports.

10

Using the Categories Node

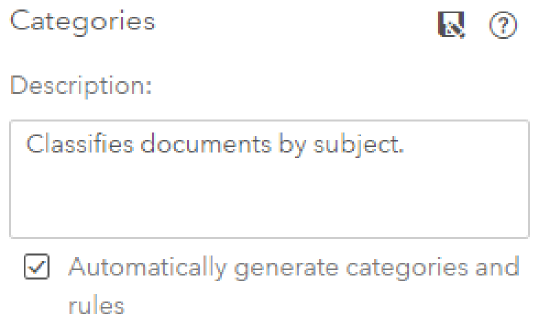
Overview

A *category* identifies a group of documents that share a common characteristic. The **Categories** node enables you to create categories using different methods, which are described in the following sections. For more information about the **Categories** node, see the following:

- “Specifying Settings for the Categories Node” on page 69
- “Using the Interactive Window for the Categories Node” on page 70
- “Using the Results Window for the Categories Node” on page 73

Specifying Settings for the Categories Node

By default, SAS Visual Text Analytics can automatically generate categories and rules for topics that are added as categories, as well as for variables that are designated as category variables in the **Data** tab. However, you can deselect the **Automatically generate categories and rules** option to save processing time if you are writing category rules yourself.



Categories

Description:

Classifies documents by subject.

☑ Automatically generate categories and rules

You must run the **Categories** node in order to see any automatically generated categories and their rules.

Using the Interactive Window for the Categories Node

Creating Categories from Category Variables

During project creation, you can assign the **Category** role to variables that you want to use for categorical analysis. When you run a **Categories** node, a new category is created for each category variable, along with a set of rules that are automatically generated.

Note: Rules might not be generated for every value in a variable with the Category role. This is because rules are generated only if they show a statistically significant relationship between specific terms and the category value. In some cases, the terms might not occur frequently enough to pick up this significant relationship.

In order to create a category using a category variable, complete the following steps:

1. Navigate to the **Data** tab, and select a variable from the variables table.
2. Locate the **Role** field in the upper right corner of the **Data** tab, and select **Category** from the drop-down list.
3. Navigate to the **Pipelines** tab, and select the **Categories** node.
4. Locate the options panel for the **Categories** node on the right side of the **Pipelines** tab, and select the **Automatically generate categories and rules** option if it is not already selected.

   Note: A **Text Parsing** node should precede the **Categories** node when automatically generating categories and rules.
5. Right-click on the **Categories** node, and select **Run**.
6. Once the pipeline runs successfully, right-click the **Categories** node and select **Open**.

The category variable is displayed in the **Categories** panel. When you select a category in the **Categories** panel, you can see the rules generated for that category in the **Edit Category** panel.

Creating Custom Category Rules

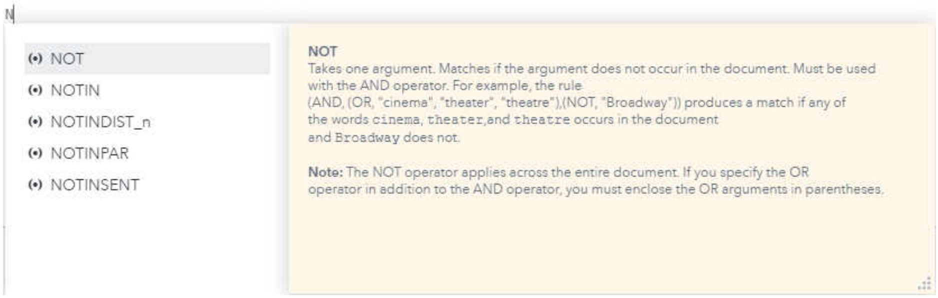
You can create custom categories by writing your own category rules. In order to create a custom category, complete the following steps:

1. Navigate to the **Pipelines** tab.
2. Right-click on the **Categories** node, and select **Open**.

3 Right-click **All Categories** in the upper left corner of the page, and select **Add new category**. The Add Category window appears.

4 Enter a name for the new category, and click **OK**. Once the new category is created, you are directed to the **Edit Category** panel.

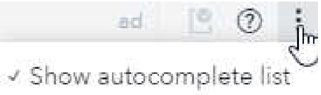
5 Create category rules for the new category, using the **Edit Category** panel to create category rules for the new category. For more information about writing category rules, see “Writing Category Rules: Boolean Rules” on page 102.



NOT
Takes one argument. Matches if the argument does not occur in the document. Must be used with the AND operator. For example, the rule (AND, (OR, "cinema", "theater", "theatre"),(NOT, "Broadway")) produces a match if any of the words `cinema`, `theater`,and `theatre` occurs in the document and `Broadway` does not.

Note: The NOT operator applies across the entire document. If you specify the OR operator in addition to the AND operator, you must enclose the OR arguments in parentheses.

If you want to disable this feature, click ⋮ and deselect **Show autocomplete list**.



6 When you are finished creating your category rules, click ☑ in the **Edit Category** toolbar to validate your new category rules.

7 Once your category rules have been validated, click **Run Node** in the upper right corner of the page to create the new category.

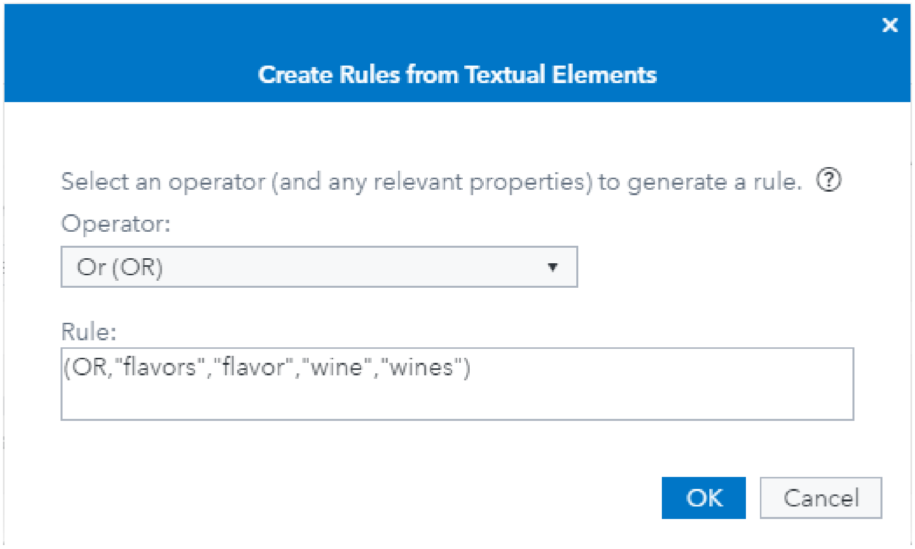
Creating Categories from Textual Elements

The **Textual Elements** pane contains the terms that were kept during text parsing, and therefore is identical to the **Kept Terms** panel in the interactive window for the **Text Parsing** node. You can use the **Textual Elements** panel to create a rule for an existing category, or to create a rule for a new category. To create a rule from the **Textual Elements** panel, complete the following steps:

1 Select a category from the **Categories** panel.

2 Locate the **Textual Elements** panel, and select the terms that you want to use in your category rule.

3 In the upper right corner of the **Textual Elements** panel, click ✱. The Create Rules from Textual Elements window appears.


Create Rules from Textual Elements

Select an operator (and any relevant properties) to generate a rule.

Operator: Or (OR)

Rule: (OR,"flavors","flavor","wine","wines")

4 Select an operator from the drop-down list in the **Operator** field, and click **OK**. The new category rules are created for the selected category.

Note: The new rule replaces any previous rule associated with the selected category.

5 Click **Run Node** in the upper right corner of the page.

Note: When you create category rules from textual elements, you do not need to validate the code before running the node.

View Matching Documents by Category

The **Documents** tab consists of an **All** tab and a **Matched** tab. In order to display matching documents for a category, complete the following steps:

1 In the **Pipelines** tab, right-click the **Categories** node and click **Open**.

2 Select a category from the **All Categories** list.

Note: Selecting a category that contains child categories will not return any matches.

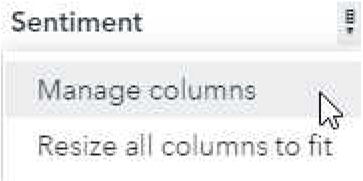
3 Click the **Matched** tab in the **Documents** tab.

If a parent category is selected for matching, and that category has a child category, matches are shown only for the parent category. If you want to see matching documents for a child category, you must select the child category. The highlighted terms are the terms that determined the document's membership in the category.

Note: In the case that emoji characters are present in the data source, they are rendered as a diamond character with a ? in it within Model Studio.

When matches are returned, you can search within the set of returned documents by creating custom syntax in the search bar. For information about using search syntax, see "Performing Searches on a Document Collection" on page 77. You can also add a **Relevancy** column to the **Documents** tab, which displays a relevancy score for each matching document. A relevancy score is a score that indicates how well a document satisfies a rule or model. The best match has a score of 1 and reflects a perfect (100%) match. To add a **Relevancy** column, complete the following steps:

1 Click ⋮ in the upper right corner of the **Documents** tab, and select **Manage columns**. The Manage Columns window appears.



2 Select **Relevancy** from the **Hidden columns** list and click +› to add it to the **Displayed columns** list.

3 Click **OK** to create the new **Relevancy** column.

Using the Results Window for the Categories Node

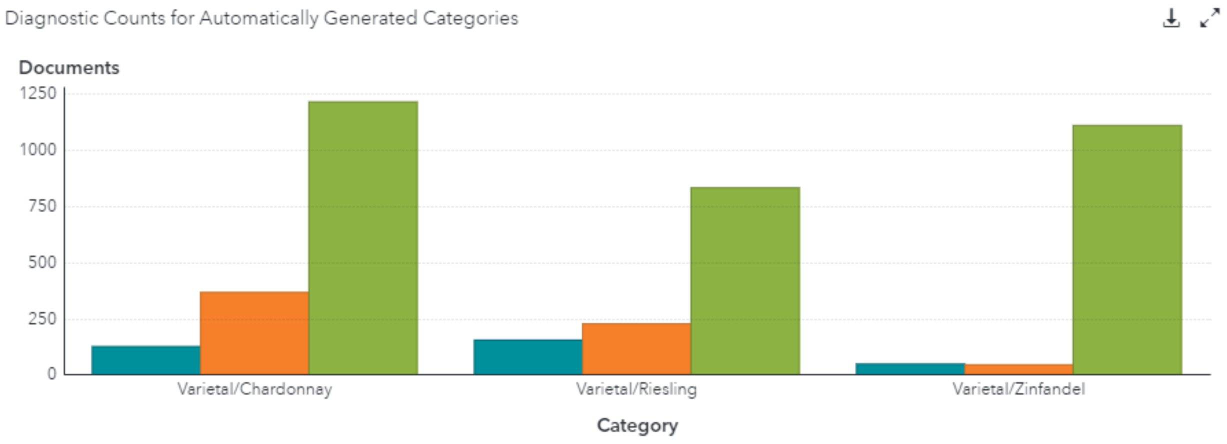
Overview

After a pipeline has run successfully, you can view results for the **Categories** node by right-clicking on the node and selecting **Results**. The Results window contains a **Summary** tab as well as an **Output Data** tab. These two tabs are explained in detail below. In some cases, they enable you to create output data that can be used for further modeling. The content and functionality within each Results window varies between node types. Features of the Results windows for each node type are explained in detail below.

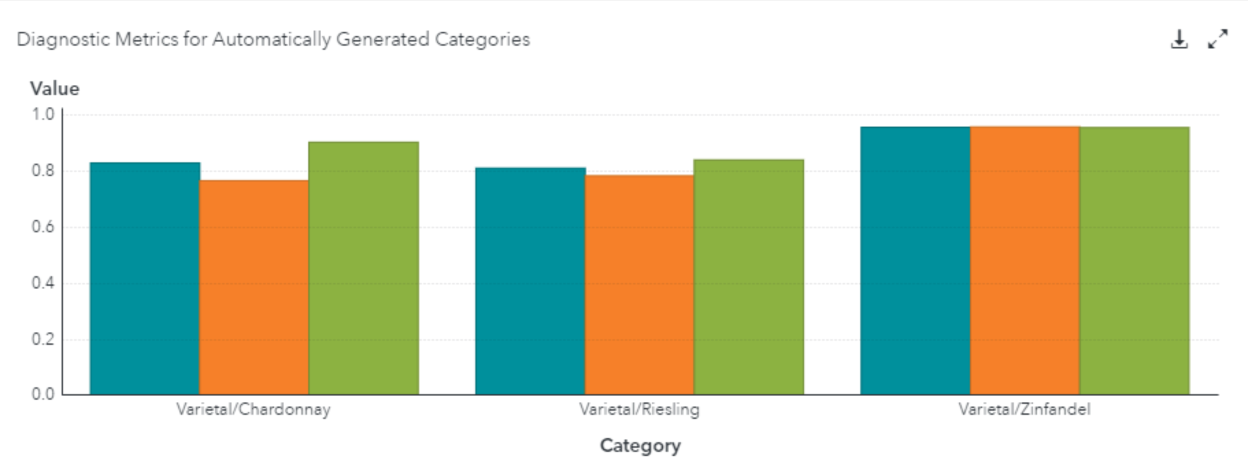
Performing Tasks in the Summary Tab

The number of components that are present in the **Summary** tab depend on the presence of automatically generated categories. If no automatically generated categories were created during the pipeline run, only the **Categories Score Code** is displayed. However, when automatically generated categories are created, graphical summaries are displayed for **Diagnostic Counts for Automatically Generated Categories** and **Diagnostic Metrics for Automatically Generated Categories**.

The **Diagnostic Counts for Automatically Generated Categories** chart shows document counts for the number of true positives, false positives, and false negatives by category.



The **Diagnostic Metrics for Automatically Generated Categories** chart displays the F-Measure, Precision, and Recall values for each automatically generated category. A lower number of false positives results in a higher precision value, and a higher number of false positives will result in a smaller precision value. The recall value is dependent upon the number of false negatives that are present. A lower number of false negatives results in a higher recall value. A higher number of false negatives results in a lower recall value. The F-Measure is a reflection of both the recall and precision values. Each of these three measures are represented by a value between 0 and 1.



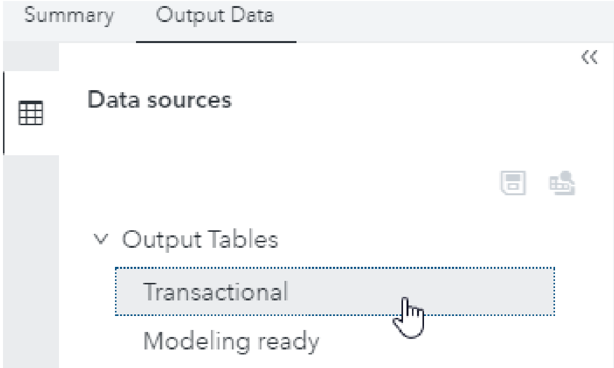
Maximizing the **Diagnostic Counts for Automatically Generated Categories** chart enables you to view the category, document count, and count type associated with each bar. Maximizing the**Diagnostic Metrics for Automatically Generated Categories** chart enables you to view the category name, decimal value, and metric type (precision, recall, or F-measure) associated with each bar. In order to see the values represented in each chart, click ⤢ in the upper right corner of either one. When you maximize the view for either chart, position your cursor over each bar to see the values represented by each one.

You can also download the data from each graph by clicking ⤓ in the upper right corner of either one. The resulting output is a CSV file.

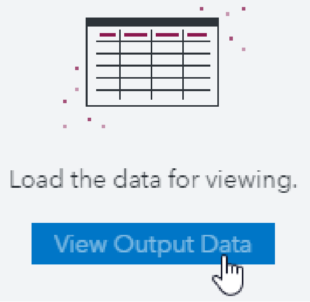
Performing Tasks in the Output Data Tab

The **Output Data** tab enables you to generate both **Transactional** and **Modeling ready** output tables. In order to create an output table, complete the following steps:

1 Locate the **Data sources** panel in the upper left corner of the **Output Data** tab, and select the desired output table type from the **Output Tables** list.



2 Click **View Output Data** to load the data.


Load the data for viewing.

When the creation of the output table is complete, the table automatically appears. In order to save your output table, click the 🖫 icon in the upper left corner of the **Output Data** tab.

If you want to visually explore your output data, complete the following steps:

1 Click in the upper left corner of the **Output Data** tab. The Explore and Visualize Output Data window appears, prompting you to choose a CAS library to save the output data.

2 Select a data source, and click **Explore and Visualize** in the lower right corner of the Explore and Visualize Output Data window.

This will redirect you to SAS Visual Analytics, where you can use a variety of tools to model your data. For information about using SAS Visual Analytics, see SAS Visual Analytics: Getting Started with Reports.

11

Exploring the Document Collection

Performing Searches on a Document Collection

The search feature in the documents panels of the interactive windows for each node can help you refine your document collection. You can also search matching documents, giving you the power to fine-tune results. Document panels are present in the interactive windows for the Concepts, Text Parsing, Topics, and Categories nodes.

In order to search a document collection, place your cursor inside the search bar in the **Documents** panel. Use the operators below in conjunction with your search query to create a more effective search.

- Place a `+` in front of a term to find documents containing that term. For example, to find all documents that contain the words *furniture* and *leather*, type **`+furniture +leather`** into the search bar. If a search query contains a term that does not have a `+` in front of it, then that term is considered optional. For example, the query `+furniture leather` returns all documents that contain the term *furniture*, and highlights the term *leather* if it is present.

- Place a - in front of a term to find documents that do not contain that term. For example, to find all documents that do not contain the word *leather*, type `-leather` into the search bar. When using only the - operator with a term in the search bar, matches are not highlighted as the term is not present in matching documents. However, the number of documents is updated to show only those that do not contain the queried term.

  Note: If you use the - operator to search on data that includes empty documents, those documents will not be included in the search results if there is a preceding **Sentiment** node.

- Place a `~` in front of a term to find documents that contain either that term or a child term. For example, entering `+~include` into the search bar returns documents that contain either the parent term, *include*, a child term (such as *includes*), or both the parent term and a child term. You can also place a `~` between the - operator and a term, which returns matches on documents that do not contain the specified term nor any of its child terms. The `~` operator returns only child terms for nodes that are preceded by a Text Parsing node. If a term is being used in conjunction with the `~` operator in a search query, and that term does not exist, the `~` operator is stripped from the query.

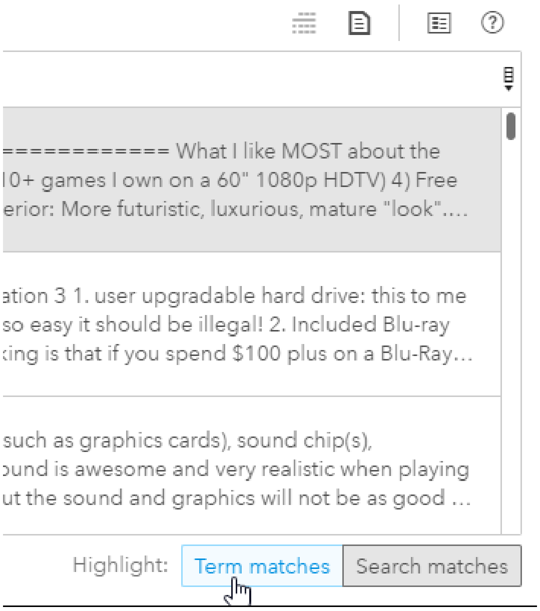
- Place a `*` at the beginning, in the middle, or at the end of a search query to return matches on wildcards. Placing a wildcard at the beginning of a search query returns matches on terms that end in the queried string of text. For example, the query `+*ion` would return documents containing terms such as *exception* or *action*. If a wildcard operator is placed in the middle of a search query, matches are returned on words that start with the text string in front of the wildcard and end with the text string after the wildcard. For example, the query `+se*e` would return matches for documents containing words like *separate*, *service*, and *seize*. If a wildcard operator is placed at the end of a search query, matches are returned on terms that start with the specified text string. For example, the search query `+comp*` returns documents that contain terms such as *complaint*,*compare*, and *compromise*.

Note: The `*` operator cannot be used in conjunction with the ~ operator. This is because the ~ operator treats the `*` symbol as a literal as opposed to an operator.

- Place quotation marks around queries when searching for multi-word terms or for a specific string of text. For example, the search query `"lost bag"` will return all documents that contain the text string *lost bag*. The search query `lost bag`, which does not contain quotation marks, returns all documents that contain either the term *lost* or *bag*.

You can use search queries to further refine a set of matching documents for terms, categories, topics, or concepts. For example, if your corpus contains 2000 documents, and only 500 of those documents are returned as matches for a selected entity, then the **Matched** tab is updated and displayed as **Matched (500 of 2000)**. When the set of matching documents for a selected entity (term, topic, concept, or category) is returned, enter your query in the search bar next to the **Matched** button and click the 🔍 icon. If 250 of the original 500 matching documents match your search query, the **Matched** button is updated to show **Matched (250 of 500)**.

By default, the matches for your search query are highlighted. However, to highlight matches for the entity that you originally selected, you can use the toggle button in the bottom right corner of the **Documents** panel.



============ What I like MOST about the
l0+ games I own on a 60" 1080p HDTV) 4) Free
erior: More futuristic, luxurious, mature "look"....

ation 3 1. user upgradable hard drive: this to me
so easy it should be illegal! 2. Included Blu-ray
:ing is that if you spend $100 plus on a Blu-Ray...

such as graphics cards), sound chip(s),
ound is awesome and very realistic when playing
ut the sound and graphics will not be as good ...

Note: The image above is from the **Documents** panel within the interactive window for the Text Parsing node. **Documents** panels in the interactive windows for the Concepts, Topics, and Categories nodes have a toggle button for **Concept matches**, **Topics matches**, and **Category matches**, respectively.

Using the Filter and Similarity Scoring Features

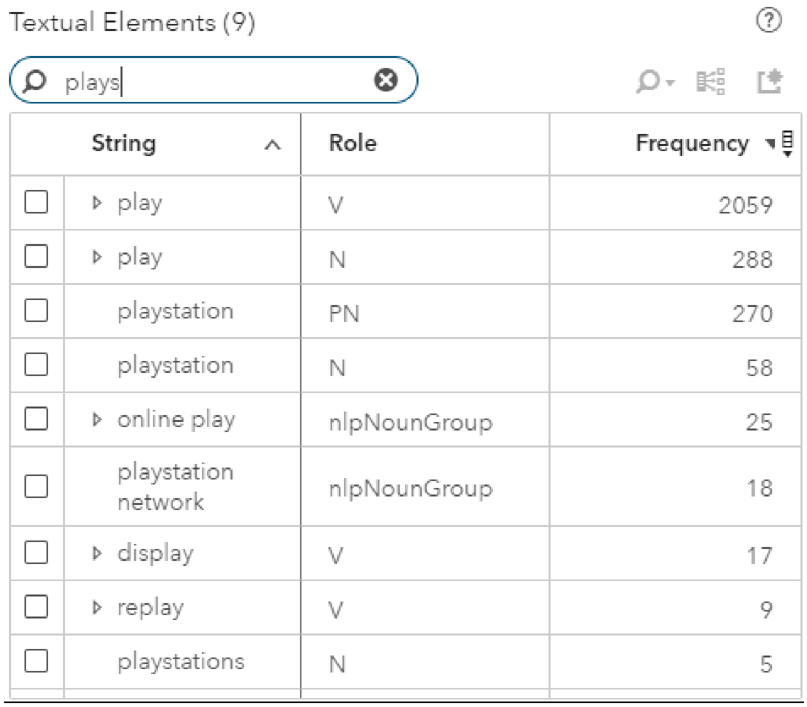
Filtering Terms

Terms can be filtered in the **Terms** panel, **Kept Terms** panel, **Dropped Terms** panel, and **Textual Elements** panel. Filtering works by returning any terms that contain the text in your filter query, which means both partial matches and exact matches are returned. As you make changes in the **Filter** bar, the list of terms being returned is automatically updated to reflect each modification made to your filter query. When viewing the matching terms that are returned, you might notice that not all of them contain the text string that you entered in the **Filter** bar. There are two circumstances that will cause this behavior:

- When a parent term matches a filter query, all of its child terms are returned with it regardless of whether they match the queried text.
- When a child term matches a filter query, its associated parent term and any other child terms of the associated parent term are returned.

The example below describes the match types that you can expect when using the **Filter** bar.

The **Textual Elements** panel shown below shows the results that are returned when the filter query *plays* is used.



Textual Elements (9)

plays

	String	Role	Frequency
□	▷ play	V	2059
□	▷ play	N	288
□	playstation	PN	270
□	playstation	N	58
□	▷ online play	nlpNounGroup	25
□	playstation network	nlpNounGroup	18
□	▷ display	V	17
□	▷ replay	V	9
□	playstations	N	5

Below is an explanation of some of the matches found for this particular filter query.

- The term *play* is returned as a match because the filter term, *plays*, is a child term of *play*. This means that all other child terms of *play* are returned as well.
- The terms *playstation*, *playstation network*, and *playstations* are returned because they contain the filter term, *plays*.
- The terms *online play*, *display*, and *replay* are returned because they have child terms that match the filter. The respective matching child terms are *online plays*, *displays*, and *replays*.

Filtering terms is a quick yet effective way to get a grasp on the contents of your document collection, and can help you develop more robust concept and category rules. For example, suppose the category **`XboxUsers`** is defined by the simple rule `(AND,"xbox",(OR,"play","use","gamer"))`. Although this rule returns relevant documents, it is a very simple rule, and therefore might fail to return many other documents that are also relevant.

Using the filter bar, you can identify other terms that are relevant to rules that you want to create. Using the category rule for **`XboxUsers`**, suppose you filter on the terms from that rule. These terms are *xbox*, *play*, *use*, and *gamer*. The results for each filter query are as follows:

- For the filter query *xbox*, the terms *xbox360*, *xbox system*, and *xbox console* are returned.
- For the filter query *play*, the term *player* is returned.

- For the filter query *use*, the term*user* is returned.
- For the filter query *gamer*, the terms *hardcore gamer* and *casual gamer* are returned.

From the results that are returned for each filter query, the category rule is modified as follows: `(AND, (OR,"xbox","xbox360","xbox system","xbox console"), (OR,"play","player","use","user","gamer","hardcore gamer","casual gamer"))`. By using the terms that you discovered using the filtering mechanism, you create a rule that is more inclusive, resulting in a larger and more representative collection of documents.

Generating Similarity Scores

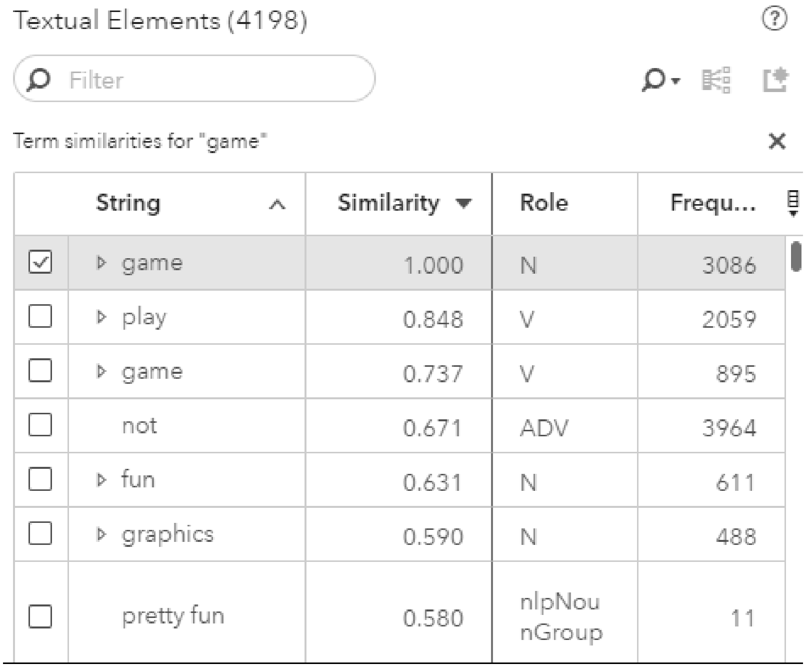
As you explore your textual data, it might be useful to know which terms are “similar” to—that is, likely to appear in the same context as—a selected term in your documents. You can generate similarity scores in the following elements:

- The **Kept Terms** panel in the **Text Parsing** node
- The **Terms** panel in the **Topics** node
- The **Textual Elements** panel in the **Categories** node and the **Concepts** node.

  Note: In order to generate a **Textual Elements** panel in a **Categories** node or a **Concepts** node, you must have a preceding **Text Parsing** node.

Understanding which terms appear in similar contexts can be useful for creating category rules, concept rules, and user-defined topics. Although the following directions show you how to generate similarity scores in a **Textual Elements** panel, the same steps are used to generate similarity scores in the **Kept Terms** panel and the **Terms** panel. In order to generate similarity scores in a **Textual Elements** panel, complete the following steps:

1. Select a term from the **String** column.
2. Click in the upper right corner of the **Textual Elements** panel to generate similarity scores for the selected term.



Textual Elements (4198)

Filter

Term similarities for "game"

	String	Similarity	Role	Frequ...
☑	▹ game	1.000	N	3086
□	▹ play	0.848	V	2059
□	▹ game	0.737	V	895
□	not	0.671	ADV	3964
□	▹ fun	0.631	N	611
□	▹ graphics	0.590	N	488
□	pretty fun	0.580	nlpNounGroup	11

Larger similarity scores indicate that a term is more likely to appear in the same context as the selected term. A score of 1.0 is an exact match (in other words, the term itself). In order to hide similarity scores, click ✕ in the upper right corner of the **Textual Elements** or **Terms** panel.

12

Writing Rules

Writing Concept Rules: Basic LITI Syntax

Introduction to Concept Rules

Concept rules are written using LITI (language interpretation for textual information) syntax. Concept rules recognize items in context so that you can extract only the pieces of the document that match the rule. For example, you can create a custom concept named **`LaGuardiaAirportComments`**, and then write a rule that extracts all documents in your document set that contain the word **`LGA`**. In other words, all of the documents displayed for the concept **`LaGuardiaAirportComments`** would contain **`LGA`**.

Each document is evaluated separately for matches. Matches do not span documents.

For information about editing rules by using the interface and by using properties settings, see For a list of rule types, see "Which Rule Type Should I Use?" on page 84.

The following list provides basic guidelines for using LITI syntax to write concept rules. The syntax is flexible, and therefore the syntax elements can be combined in numerous ways.

- A rule consists of a rule type (which is written in uppercase letters), followed by a colon, then by arguments. For example, in the rule **`CLASSIFIER:LGA`**, **`CLASSIFIER`** is the rule type, **`LGA`** is the argument, and they are separated by a colon. Rule modifiers can be used to further refine the set of matches. The rule syntax varies greatly depending on the rule type; the basic syntax is included in the description of each concept rule in

Table 12.1 on page 85, and each fact rule in Table 12.2 on page 87. For a list of rule modifiers, see "Adding Rule Modifiers" on page 88.

- Use descriptive concept rule names that cannot be used as single words (for example, baseballScore). You can also include information about how you will use the concept in other rules by using a prefix (for example, Helper_BaseballScore).
- A single concept rule can reference one or more other concept names. You can also write rules that recognize key words or elements within a specific context. For example, you can extract documents that contain the string **LGA** only if it appears before the word **Airport**.
- Use part-of-speech tags in rules to identify linguistic structures. For more information, see "Using Part-of-Speech and Other Tags" on page 95.
- Use Boolean and proximity operators to enhance the precision of your rules. For more information, see "Using Boolean Operators for Extracting Concept Rules and Facts" on page 90.
- Use morphological expansion operators to return inflected forms of a word.
- Use coreference operators to resolve pronouns. For example, if the pronoun **he** were used to refer to **Walt Disney**, you can write a rule that specifies the canonical form (full form) and returns it in the concept. For more information, see "Using the Coreference Operator" on page 94.

Concepts versus Facts

Facts (also called predicates) are related pieces of information in text that are located and matched together.

Facts can be identified within a custom concept. For example, suppose you want to identify US universities that were named after presidents. You could write a rule that identifies **George Washington** as a US president (**US_President_Names**) and also identifies **George Washington University** as a university named for him (**UNIVERSITY**).

So, in the sentence **There are countless active student organizations at George Washington University**, the string **George Washington** would match the concept **US_President_Names** and **George Washington University** would match **UNIVERSITY**.

You can use the following special types of concept rules to locate facts:

- A predicate rule (PREDICATE_RULE) uses Boolean and proximity operators to help locate facts. For example, you can use Boolean and proximity operators to specify terms that you want to occur within a certain number of terms of each other. The following rule identifies occurrences of the term **America** (denoted as **country**) that occurs within three terms of **flag**, **emblem**, or **crest**:

  ```
  PREDICATE_RULE:(country):(DIST_3,"_country{America}",

  (OR, "flag", "emblem", "crest"))
  ```

- You can use a sequence rule (SEQUENCE) when the order of the items in the fact is important. A sequence rule can detect a structure so that each term in the fact matches in the order that you specify with no intervening items.

Which Rule Type Should I Use?

There are several distinct types of rules for extracting concepts and facts. You can specify more than one rule in each custom concept or fact. It is important to understand the rule types so that you can select those that efficiently generate the most matches for your purposes.

Note: For the concept rule syntax listed in the following tables, < > denotes an optional syntax element. Items in *italics* denote values that you must supply, such as strings and concept names.

The table below lists the types of rules that are used for extracting concepts. Included is a brief description of how each rule type is used, along with basic syntax. For examples of concept rule syntax, see "Concept Rule Types: Examples" on page 101.

Table 12.1 *Overview of Rules for Extracting Concepts*

Rule Type	Description and Basic Syntax
CLASSIFIER	Identifies single terms or strings that you want matched in context. For example, in a concept definition, you can create CLASSIFIER rules that contain specific airport codes. The portions of text that contain the airport codes are considered matches to the CLASSIFIER rules. `CLASSIFIER:string<,information>` When you want to match the character # as part of a CLASSIFIER rule argument, you must precede it with the character \. When you want to match the character , as part of a CLASSIFIER rule argument, you must use the character combination `\c`. For example, the sentence `Stop, drop, and roll.` would be returned as a match for the rule `CLASSIFIER: Stop\c drop\c and roll.`
CONCEPT	Identifies related information by referencing other concepts. For example, to capture documents that contain certain US airport names and locations, you can create a CONCEPT rule type in the definition. The CONCEPT rule type can reference any other concept. For example, it can reference a concept that contains a list of CLASSIFIER rules defining airport codes, thereby accessing a list of airport codes. CONCEPT is a rule type. It is not to be confused with a "concept" in the general sense. **Note:** The concept that you are referencing in the rule is also matched as a string. For example, in the rule **`CONCEPT:SCORE`**, the string **`SCORE`** is matched. Therefore, it is recommended that you use concept names that cannot be used as single words (for example, baseballScore). `CONCEPT:argument-1<argument-n>` where *argument* can be a concept name, rule modifier, or string.
C_CONCEPT	Returns matches that occur in the specified context only. For example, to extract matches that include names of university professors, you could create a C_CONCEPT rule that identifies matches on a concept (previously defined) that identifies last names only when the matched names are preceded by the word **`Professor`**. **Note:** This rule type requires the **`_c{}`** modifier. `C_CONCEPT:<argument>_c{argument}<argument>` where *argument* can be a concept name, rule modifier, or string.

Rule Type	Description and Basic Syntax
CONCEPT_RULE	Uses Boolean and proximity operators to determine matches. For a list of operators, see “Using Boolean Operators for Extracting Concept Rules and Facts” on page 90. **Note:** This rule type requires the **`_c{}`** modifier. Quotation marks (") must surround the strings that you want to match. The **`_c{}`** can surround only one argument, which is highlighted when matches are returned. The other arguments that appear in quotation marks provide context for the match and must be present for a match to occur. `CONCEPT_RULE:(<Boolean-rule-1>...<Boolean-rule-n>` where *Boolean-rule* can be a nested *n* times, and is written as: `Boolean-operator"_cargument-1}",<"argument-2">...<"argument-n">)`
NO_BREAK	Prevents partial matches by ensuring that a match occurs only if the entire string is located. For example, suppose you want to capture text that includes the item **`National Gallery of Art`**. You can create a rule that ensures that the entire string **`National Gallery of Art`** is matched and not **`Gallery`** and **`Art`** as separate items. When using NO_BREAK, remember the following: ■ This rule type requires the **`_c{}`** modifier. ■ NO_BREAK applies across the entire taxonomy regardless of where the rule appears or whether the rule is enabled or disabled. ■ Do not insert NO_BREAK rules just anywhere. It is helpful to insert them all in one concept. That is, create a concept that contains globally implemented rules only (NO_BREAK or REMOVE_ITEM). Having such rules all in one place aids in troubleshooting the matching behavior across your taxonomy. `NO_BREAK:_c{argument}` where *argument* can be a concept name (not recommended) or a string.
REGEX	Identifies patterns of information that can be represented as a series of character types, as in telephone numbers, ZIP code, product numbers, or hyphenated words. No other elements can be placed in a REGEX rule with the exception of the regular expression itself. Also, the boundaries of the match must coincide with token boundaries; you cannot match a partial token with a REGEX rule. For example, `REGEX:[0-9]{5}` matches any five digit number to help find ZIP codes in the USA. `REGEX:`*regular-expression*

Rule Type	Description and Basic Syntax
REMOVE_ITEM	Ensures that a correct match is made when one word is a unique identifier for more than one concept. For example, you can write a rule that distinguishes between the Arizona **`Cardinals`** football team and the St. Louis **`Cardinals`** baseball team. The context of each match is used to eliminate incorrect matches. Note: This rule type requires the **`_c`** modifier and the ALIGNED operator. Quotation marks (") must surround the strings that you want to match. `REMOVE_ITEM:(ALIGNED, "_c{concept name}",<"argument">` where *argument* can be a concept name or a string.

Table 12.2 on page 87 lists the rules used for extracting facts. Included is a brief description of how each rule type is used, along with basic syntax.

Table 12.2 Overview of the Rules for Extracting Facts

Rule Type	Description and Basic Syntax
PREDICATE_RULE	Helps you define facts that you want identified in text. For information about facts, see "Concepts versus Facts" on page 84. `PREDICATE_RULE:(argument-name-1...<argument-name-n>):(Boolean-rule-1...<Boolean-rule-n>)` where *argument-name* refers to a name that you specify for fact matching, and where *Boolean-rule* can be nested *n* times, and is written as: `(Boolean-operator,"_argument-name{argument}",..."<_argument-name>{<argument>}")` The PREDICATE_RULE rule type is more flexible than the SEQUENCE rule type because it does not always specify order.
SEQUENCE	Identifies facts in documents if the facts appear in the order specified with no intervening elements. For information about facts, see "Concepts versus Facts" on page 84. `SEQUENCE:(argument-name-1...<argument-name-n>):_argument-name-1{argument}<_argument_name_n{argument}>` where *argument_name* refers to a name that you specify for fact matching, and where *argument* can be a concept name, rule modifier, or string. Note: This syntax is written in its simplest form. Additional modifiers and arguments for concept rule matching can be inserted. The SEQUENCE rule type requires the number of *argument-names* specified must match the number of *_argument-names* applied.

Using Punctuation

Use punctuation to qualify the matches for all rule types except CLASSIFIER and CONCEPT.

Table 12.3 Punctuation in CLASSIFIER and CONCEPT Rule Types

Type of Punctuation	Description
Colon	Separates rule types and tags. Use a colon under the following circumstances: ■ After a concept rule type (for example, **`CLASSIFIER:`**) ■ Between the arguments list and the SEQUENCE or PREDICATE_RULE definition. ■ Before a part-of-speech tag (for example, **`:Prep`**).
Comma	Separates operators and arguments in a CONCEPT_RULE or PREDICATE_RULE definition. Add a space after the comma and before the next argument.
Single Space	Separates strings, concepts, part-of-speech tags, and rule modifiers in CONCEPT, CONCEPT_RULE, SEQUENCE, and C_CONCEPT rule types.
Quotation Marks	Encloses concept names and strings in arguments for CONCEPT_RULE, REMOVE_ITEM, and PREDICATE_RULE rule types.
Parentheses	Groups the arguments with each operator in CONCEPT_RULE, REMOVE_ITEM, SEQUENCE, and PREDICATE_RULE rule types.
Square Braces	Character class in the REGEX rule type.
Curly Braces	Delimits information that is returned as a match.

Adding Rule Modifiers

Several types of concept rule modifiers can enhance the matching ability of a rule. The following tables list the types of rule modifiers available, and denote which rule types they can be used in.

Table 12.4 Concept Rule Modifiers and Associated Rule Types

Modifier	CLASSIFIER	CONCEPT	C_CONCEPT	CONCEPT_RULE
Comments	X	X	X	X
Context (_c{})			X (Required)	X (Required)
Word (_w)		X	X	X
Word with initial capital letter (_cap)		X	X	X
Multiple matches symbol (>)			X	X

Modifier	CLASSIFIER	CONCEPT	C_CONCEPT	CONCEPT_RULE
Morphological expansion symbols (@, @A, @N, and @V)		X	X	X
Boolean and proximity operators				X
Part-of-speech tags		X	X	X
Export feature	X			
Coreference symbols (_ref{}, _P, and _F)		X	X	X
Regular expressions (Regex)				
Predefined concepts		X	X	X

Table 12.5 Concept Rule Modifiers and Associated Rule Types, Continued

Modifier	REMOVE_ITEM	NO_BREAK	SEQUENCE	PREDICATE_RULE	REGEX
Comments	X	X	X	X	
Context (_c{})	X (Required)	X (Required)			
Word (_w)	X	X	X	X	
Word with initial capital letter (_cap)	X	X	X	X	
> symbol					
Morphological expansion symbols (@, @A, @N, and @V)	X	X	X	X	
Boolean and proximity operators				X	
Part-of-speech tags	X	X	X	X	
Export feature					
Coreference symbols (_ref{}, _P, and _F)					
Regular expressions (Regex)					X (Required)

Modifier	REMOVE_ITEM	NO_BREAK	SEQUENCE	PREDICATE_RULE	REGEX
Predefined concepts	X	X	X	X	

Using Boolean Operators for Extracting Concept Rules and Facts

The table below lists Boolean operators that you can use when you write concept rules and identify facts.

Table 12.6 Boolean Operators for Extracting Concept Rules and Facts

Operator	Description
ALIGNED	Takes two arguments, where an argument is either a set of elements specified within a set of double quotation marks, or an operator and its arguments. Returns a match when both arguments have the same matching span of text in a document. Used with the REMOVE_ITEM rule type only. For example, the following rule says to remove the match for the concept **`DATE`** if that match is followed by the word *driver*, and matches the string *Sunday driver*. This ensures that *Sunday driver* will not return as a match for **`DATE`**. `REMOVE_ITEM:(ALIGNED, "_c{DATE} driver", "Sunday driver")`
AND	Takes one or more arguments. Matches if all arguments occur in the document, in any order. For example, the following rule returns a match on **`King Louis XIV`** if it occurs in the document with **`France`**: `CONCEPT_RULE:(AND, "_c{King Louis XIV}", "France")`

Operator	Description
DIST_*n*	(Distance) Takes a value for *n* and two or more arguments. Matches if all arguments occur within *n* (or fewer) tokens of each other, regardless of their order. If an argument contains multiple tokens, then distance is calculated from the first token of the first argument to the last token of the last argument. **Note:** The `DIST_` operator does not calculate distance for concept rules in the way it calculates distance for category rules. For example, the following rule returns a match in the phrase **`standard contract for the supply of goods`**: `CONCEPT_RULE:(DIST_6, "_c{standard contract}", "for the supply", "of goods")` **Note:** For calculation purposes, the distance between tokens is not inclusive. For example, the distance between **`best`** and **`show`** in the phrase **`best in show`** is two tokens. Tokens that include hyphens are counted as one (for example, **`merry-go-round`** is one token).
NOT	Takes one argument. Matches if the argument does not occur in the document. Must be used with the AND operator. For example, the following rule returns a match if **`cinema`**, **`theater`**, or **`theatre`** occur in the document, but **`Broadway`** does not: `CONCEPT_RULE: (AND, (OR, "_c{cinema}", "_c{theater}", "_c{theatre}"), (NOT, "Broadway"))` **Note:** The NOT operator applies across the entire document. All operators must have their own parentheses around themselves and their associated arguments.
OR	Takes one or more arguments. Matches if at least one argument occurs in the document. For example, the following rule returns a match if one or more of the items **`U.S.`**, **`US`**, or **`United States`** appear in the document: `CONCEPT_RULE:(OR, "_c{U.S.}", "_c{US}", "_c{United States}")` **Note:** Rules that are generated by SAS Visual Text Analytics nest the OR operator within the AND operator. However, the OR operator can stand alone.
ORD	(Order) Takes one or more arguments. Matches if all of the arguments occur in the order specified in the rule. For example, the following rule returns a match in the sentence **`The warranty claim for the washing machine was denied.`**: `CONCEPT_RULE:(ORD, "warranty", "claim", "denied")`

Operator	Description
ORDDIST_*n*	(Order and distance) Takes a value for *n* and two or more arguments. Matches if all arguments occur in the same order that is specified in the rule and if all arguments are within *n* tokens of each other. When arguments contain multiple tokens, the distance is calculated from the first token of the first argument to the last token of the last argument. **Note:** The `ORDDIST` operator does not calculate distance for concept rules in the way it calculates distance for category rules. For example, the following rule returns a match in the phrase **`standard contract for the supply of goods`**: `CONCEPT_RULE:(ORDDIST_6, "_c{standard contract}", "for the supply", "of goods")` **Note:** For calculation purposes, the distance between tokens is not inclusive. For example, the distance between **`best`** and **`show`** in the phrase **`best in show`** is two tokens. Tokens that include hyphens are counted as one (for example, **`merry-go-round`** is one token).
PARA	(Paragraph) Matches if all the arguments occur in a single paragraph, in any order. For example, the following rule returns a match if the paragraph contains the term **`Manhattan`** and also includes the token **`apartment`** (Only **`Manhattan`** is highlighted.): `CONCEPT_RULE:(PARA, "_c{Manhattan}", "apartment")` **Note:** PARA rules work properly only when they are applied to data sets that contain paragraph delimiters \n\n (new line), \t\t (tab), or <P> (paragraph). PARA cannot be applied on the **Test Sample Text** tab. PARA also cannot be applied to data that is contained in folders.
SENT	(Sentence) Takes two or more arguments. Matches if all the arguments occur in the same sentence, in any order. For example, the following rule returns a match when **`Amazon`** and **`river`** occur within the same sentence: `CONCEPT_RULE:(SENT, "_c{Amazon}", "river")` Delimiters are used for sentence tokenization. Tokenization is a process that breaks up sentences into words, phrases, symbols, or other meaningful elements (tokens). Note that a period (.) does not necessarily indicate an end of sentence (for example, **`Mr. Quackenbush`** or **`Boston, Mass.`** could occur in the middle of a sentence). For a list of sentence delimiters, see Table 12.11 on page 107.

Operator	Description

SENT_*n*	(Multiple sentences) Takes a value for *n* and two or more arguments. Returns matches within *n* sentences. For example, the following rule returns a match for the concept **`GENDER`** and the term **`he`** within two sentences. Suppose the concept **`GENDER`** contains the following rule:
	`CLASSIFIER:male`
	You can then write this rule:
	`CONCEPT_RULE:(SENT_2, "_c{GENDER}", "he")`

SENTEND_*n*	(End of sentence) Takes a value for *n* and one or more arguments. Returns matches within *n* tokens of the end of the sentence. For example, suppose the concept **`GENDER`** contains the following rule:
	`CLASSIFIER:female`
	Then the following rule returns a match for the concept **`GENDER`**, and the term **`she`** within five tokens from the end of a sentence:
	`CONCEPT_RULE:(SENTEND_5, "_c{GENDER}", "she")`
	Note: When you specify the value of *n*, consider that the end of the sentence is `0`. Tokens that include hyphens are counted as one (for example, **`merry-go-round`** is one token).

SENTSTART_*n*	(Start of sentence) Takes a value for *n* and one or more arguments. Returns matches within *n* tokens of the beginning of the sentence. For example, the following rule locates matches for the sentence **`The patient experienced breathing difficulty.`**: `CONCEPT_RULE:(SENTSTART_5, "_c{patient}", "breathing", "difficulty")` **Note:** When you specify the value of *n*, consider that the beginning of the sentence is `0`. Tokens that include hyphens are counted as one (for example, **`merry-go-round`** is one token).

Operator	Description
UNLESS	Takes two arguments, the second of which is one of the following operators (with its arguments): AND, SENT, DIST, ORD, or ORDDIST. Restricts certain matches by specifying a relationship between two arguments and allowing a match only if a third argument does not intervene. Used in rule types PREDICATE_RULE and CONCEPT_RULE only. For example, the following rule does not include the token **river** in its matches. In addition, the rule returns matches for **Mississippi** the state and not **Mississippi** the river: `CONCEPT_RULE:(UNLESS, "river", (SENT, "_c{Mississippi}", "United States"))` The rule ensures that **river** does not appear between **Mississippi** and **United States** in the matches. Note: When you specify a concept governed directly by the UNLESS operator, specify concepts that contain only CLASSIFIER or REGEX rules.

Using the Coreference Operator

Use the coreference modifier (_ref{}) when you want to link pronouns and other words with the canonical form (full form) of the terms that they reference.

Suppose you have a concept named **LEADERS** that includes this rule:

```
CLASSIFIER:Congressional leaders
```

You can create the concept **THEY_SAID** that enables **they** to reference its canonical form, **Congressional leaders**. Both forms are matched in the document.

```
C_CONCEPT:_c{LEADERS} said _ref{they}
```

You can use the following symbols with the coreference modifier (_ref{}). Place the symbol after the _ref{*concept*} modifier.

- > (Multiple matches) — Locates multiple instances of a match that is specified by the coreference modifier (_ref{}). For example, you might want to return the canonical form of the name **Ms. Geraldine Jones** each time the nickname **Geri** is encountered. The > symbol enables this match to occur after the first time the canonical form of the name is located.

  ```
  C_CONCEPT:_c{Ms. Geraldine Jones} _ref{Geri}>
  ```

- _F (Forward) — Returns only matches that occur from the coreference rule match onward. Sample syntax:

  ```
  C_CONCEPT:_c{PERSON} as _ref{TITLE}_F
  ```

- _P (Preceding) — Returns only matches that occur up to and including the coreference rule match. Sample syntax:

  ```
  C_CONCEPT:_c{MILITARY BRANCH} as _ref{HONOR}_P
  ```

Using the Export Feature

The Export feature enables you to find matching occurrences of terms or phrases found in CLASSIFIER rules and then export them to one or more concepts. This feature is useful for conditional matching of terms or phrases. You can export matches from multiple concepts to one concept, or to more than one concept.

Note: The Export feature can be used only with CLASSIFIER rules.

For example, suppose you want to find all the occurrences of the term **accounts receivable** that occur together with the name **Sokolov**, and export those matches to the concept **AR**. You could write the following rule in a concept named **ACCOUNT_HOLDER**:

```
CLASSIFIER:[export=AR:accounts receivable]:Sokolov
```

The rule first matches the term **Sokolov**. If that match is found, the rule checks the documents for any occurrences of the term **accounts receivable** and assigns any matches to the concept **AR**. In the list of matches for **ACCOUNT_HOLDER**, the term **Sokolov** would be highlighted. In the list of matches for **AR**, the term **accounts receivable** would be highlighted. Note that in order for the rule to work, the primary term (in the example, **Sokolov**) needs to be present anywhere in the document before **accounts receivable** can be returned as a match for the concept **AR**.

Concepts that you are exporting to (such as **AR** in the example) must exist in the list of concepts and can contain additional rules (or be empty). The following example illustrates how to export two sets of terms to the same concept.

```
CLASSIFIER: [export=text2]:text1
```

If **text1** and **text2** appear in a document, return **text1** and **text2** as separate matches for the concept where this line is located. For example, suppose you have written the following rule:

```
CLASSIFIER:[export=SAS]:institute
```

The string **SAS institute** returns **SAS** and **institute** as matches to the concept where this line is located. The string **institute** (occurring alone) is a match, but not **SAS** occurring alone.

Using Part-of-Speech and Other Tags

Part-of-speech tags enable you to locate matches by the part of speech that the searched item belongs to, rather than locating a specific term. These tags are useful when you know the syntax but not the exact wording of an item that you are seeking. Also included are other tags that are not considered parts of speech (such as punctuation).

Because the parts of speech are sensitive to the context in which they appear, the same word might be tagged differently, depending on the surrounding text. For example, the word **will** could be tagged as a modal verb (she will be a big star someday) or noun (a last will and testament).

Part-of-speech tags are preceded by a colon (:). The tags are case-sensitive. For example, suppose you want to match an attribution for a quotation in a news article. You know that the syntax for the match appears as **Senator from** *state* or **Senator of** *state* but you do not know the name of the senator. You can use the following rule:

```
C_CONCEPT:SENATE_TITLE _c{_cap _cap} :Prep STATE
```

The rule assumes that there is a concept **SENATE_TITLE** that contains words such as **majority leader**, **senator**, and **senators**, and a concept **STATE** that includes names of states. The :Prep tag indicates a preposition (for example, **from** or **of**). A match on the C_CONCEPT rule would occur on the text **Senator Phineas Craymoor from North Carolina took the floor**. However, the following text would not produce a match because the word **and** is not a preposition: **Senators Phineas Craymoor and Garrett Garcia from North Carolina pushed the bill through**.

Table 12.7 *Part-of-Speech Tags (For English)*

Part-of-Speech Tag	Definition	Examples
:ABBREV	Abbreviation	etc., Ms, cm

Part-of-Speech Tag	Definition	Examples
:Acomp	Comparative adjective	cooler, luckier, worse
:Adv	Adverb	lyrically, physically
:Asup	Superlative adjective	mellowest, merriest, best
:C	Conjunction	when, yet, after, except
:date	Date	2000-02-21, 04/03/2012
:digit	Sequence of numbers	2345, 234.22, 21/234
:Det	Determiner	the, an, every
:F	Foreign	facto, klieg, modus
:inc	Unknown word	slaster, lijer
:Int	Interjection	hah, hello, tallyho
:Md	Modal	can, should, will
:N	Noun	cake, love, shoe
:Npl	Plural noun	peas, sheep, shoes
:Num	Number	one, twenty, hundred
:PN	Proper noun	SAS, Cary, Goodnight
:PossDet	Possessive determiner	our, his, my
:PossPro	Possessive pronoun	mine, yours, hers
:PreDet	Pre-determiner	quite, such, all
:Prefix	Prefix	cross, ex, multi
:Prep	Preposition	on, under, across
:Pro	Pronoun Relative pronoun	he, one, somebody, me myself, oneself, themselves
:Ptl	Particle	away, forward, in
:sep	Separator and punctuation	; , /
:time	Time	7AM, 10:00 pm
:url	File names, pathnames, URL	A:/mydir/file.txt, www.sas.com

Part-of-Speech Tag	Definition	Examples
:V	Undeclined *be*, *do*, or *have* auxiliary Undeclined verb First person singular verb	be, do, have go, see, love am
:V3sg	Third person singular *be*, *do*, or *have* auxiliary Third person singular verb	is, does, has goes, sees, loves
:Ving	Present participle *be*, *do*, or *have* auxiliary Present participle	being, doing, having bucketing, climbing
:Vpp	Past participle *be*, *do*, or *have* auxiliary Past participle	been, done, had dashed, factored, gone
:Vpt	Past tense *be*, *do*, or *have* auxiliary Past tense verb	was, were, did, have dashed, factored, went
:WAdv	Adverbial *wh*	how, when, whereby
:Wdet	Demonstrative determiner *wh*	which, what, whatever
:WPossPro	Possessive determiner *wh*	whose
:WPro	Nominal *wh*	whose, what, whoever

Using Regular Expressions (Regex)

Use regular expressions (Regex syntax) to identify regularly occurring patterns in the text that might include numbers, punctuation, and characters. You can use regular expressions to match patterns such as license plate numbers (example: ABX-0444), part numbers for manufacturing components (example: TMS1T3B1M5R-23), hyphenated words (example: fifty-nine), and so on. The following guidelines apply to Regex syntax:

- Characters are matched within a string in sequence when represented without square brackets (`[ ]`). For example, the following rule matches only the word **`any`** (**`anyone`** or **`anything`** would not be matched):

  ```
  REGEX:[crash]
  ```

  If you add a plus sign (`+`) as follows, the rule matches one or more of the characters specified in any combination, such as **`rash`** , **`cash`**, **`ash`**, and **`crass`** (but not **`crashpad`** or **`crashdummy`**):

  ```
  REGEX:[crash]+
  ```

- Characters are matched within a string in sequence when represented without square brackets (`[ ]`). For example, the following rule matches only the word **`any`** (**`anyone`** or **`anything`** would not be matched):

  ```
  REGEX:any
  ```

  To match words that contain **`any`**, you can modify the rule to use asterisks (*) to match other character occurrences (or none) surrounding **`any`**. For example, the following rule matches **`any`**, **`anyone`**, **`anything`**, and **`Many`**:

  ```
  REGEX:[A-Za-z]*any[A-Za-z]*
  ```

- You can specify a range of characters to be matched. For example, the following rule matches lowercase characters between **a** and **f**, inclusively:

  ```
  REGEX:[a-f]
  ```

  To add uppercase characters, use the following rule:

  ```
  REGEX:[A-Fa-f]
  ```

- You can specify characters that should not be matched (negated characters) by inserting a caret (^) before a set of characters. For example, the following rule matches all characters, numbers, and symbols in text except **a**, **e**, **i**, **o**, and **u**:

  ```
  REGEX:[^aeiou]
  ```

  Note: Matches returned by ^ are limited to ASCII characters.

- Characters that are reserved for special meaning (metacharacters) must be escaped with a backward slash (\) to be literally matched in a regular expression. The metacharacters are: **[**, **]**, **(**, **)**, ?, *****, **+**, **.**, **-**, \, and |

  For example, `[\?]` matches a question mark **?** in text.

- Numbers are matched as-is within a string when represented without square brackets (**[ ]**). For example, the following rule matches part numbers that begin with **0125-** and end with a letter:

  ```
  REGEX:0125\-[A-Za-z]
  ```

- Numbers are matched by specifying ranges when enclosed in square brackets (**[ ]**). For example, the following rule returns a match on a number between **0** and **9**:

  ```
  REGEX:[0-9]
  ```

  CAUTION! For a project whose project language is set to Korean, REGEX rules might not work as expected. Using another rule type, such as a CLASSIFIER rule, in conjunction with a REGEX rule results in the REGEX rule working as expected.

The special characters used for matching in Regex syntax can be used in combination and are shown in the table below.

Table 12.8 Special Characters (Metacharacters) Used in Regular Expressions

Character or Expression	Description
|	(Alternative) Indicates that matches occur when either regular expression *a* or *b* is matched. Example: *a* | *b*
( )	Grouping mechanism (non-remembering). Used in expressions for clarity. Example: (?:(?:*ababab*) | *b*)
.	(Wildcard) Matches any single ASCII character.
%	Matches %
?	Matches 0 or 1 occurrences
*	Matches 0 or more occurrences
+	Matches 1 or more occurrences

Character or Expression	Description
{ }	Indicates repetition: ■ {*n*} matches exactly *n* occurrences ■ {*n,*} matches at least *n* occurrences ■ {*n,m*} matches at least *n* occurrences but no more than *m* occurrences
\a	Alarm (beep)
\n	New line
\r	Carriage return
\t	Tab
\f	Form feed
\e	Escape
\d	Digit (same as `[0–9]`)
\D	Not a digit (same as `[^0–9]`)
\w	Word character (same as `[a-zA-Z_0–9]`)
\W	Non-word character (same as `[^a-zA-Z_0–9]`)
\s	Whitespace character (same as `[ \t\n\r\f]]`)
\S	Non-white-space character (same as `[^ \t\n\r\f]]`)
\xh	Hexadecimal number, where *h* is a hexadecimal character
\xhh	Hexadecimal number, where *h* is a hexadecimal character
\0o	Octal number, where *o* is an octal digit
\0oo	Octal number, where *o* is an octal digit

The following restrictions apply to Regex syntax:

- Regex syntax works similarly to regular expressions in Perl. However, the two are not identical.
- Character matching for characters, numbers, or symbols that are specified inside square brackets ([ ]) does not occur at the word level. For example, the following rule matches the isolated letters **`x`**, **`y`**, and **`z`**, but no matching occurs for the words **`xylitol`**, **`yes`**, or **`recognize`**:

  ```
  REGEX:[xyz]
  ```

  If you add a plus sign (+) to match multiple occurrences (or one occurrence) as follows, the rule matches any combination of the characters that are specified. Examples include **`xzx`**, **`yz`**, and **`zyzy`**:

  ```
  REGEX:[xyz]+
  ```

However, because of the presence of characters other than **x**, **y**, and **z**, there is no matching for words **xxl**, **syzygy**, or **diy**.

- You cannot refer to concepts in a Regex expression.
- Backward references to matches in the text are not supported.
- Parentheses **( )** as a grouping mechanism where matches are remembered are not supported. Parentheses are used merely for clarifying matching rules.

Using Morphological Expansion Symbols

You can use morphological expansion in all rule types except CLASSIFIER and REGEX. For example, to expand the word **breathe** to all verb forms, which include **breathes** and **breathing**, use the following syntax for the argument: **"breathe@V"**.

Table 12.9 Morphological Expansion Symbols in Concept Rules

Symbol	Description
@	Expands the concept rule to match all inflectional forms of the word in the argument. For example, the argument **"wonder@"** returns the matches **wonder**, **wonders**, **wondered**, **wondering**, and so on. **Note:** If you apply @ to a word that SAS Visual Text Analytics does not recognize, no expansion occurs. Only the exact string specified before the @ is matched. For example, **"grath"** would not expand. Only the string **grath** would return a match in the rule.
@A	Expands the concept rule to match inflected comparative and superlative adjective forms of the word in the argument. For example, the argument **"happy@A"** returns the matches **happier** and **happiest**. **Note:** If you apply @A to a word that is not an adjective, no expansion occurs.
@N	Expands the concept rule to match all inflected noun forms of the word in the argument. For example, the argument **"quality@N"** returns the matches **quality** and **qualities**. **Note:** If you apply @N to a word that is not a noun, no expansion occurs.
@V	Expands the concept rule to match all inflected verb forms of the word in the argument. For example, the argument **"transfer@V"** returns the matches **transfer** , **transfers**, **transferred**, and **transferring**. **Note:** If you apply @V to a word that is not a verb, no expansion occurs.

Note: Morphological expansion does not include misspellings that have been detected in the Text Parsing node.

Adding Comments

You can insert comments into rule definitions that have separate rules appearing on successive lines, such as CLASSIFIER rules. The comment continues until the end of the line. Comments are written as `#comment text`.

Note: The pound character (`#`) denotes a comment. If you want to match `#` in a rule definition, you must use a backward slash (`\`) as an escape character before the `#`. (Example: The expression `99\#` attempts to match the string `99#`.)

The pound character (`#`) can also be used to comment out a rule. To comment out a rule, insert a pound character (`#`) at the beginning of a line that contains a rule.

Concept Rule Types: Examples

Examine the syntax in the examples to understand how to write different types of concept rules.

CLASSIFIER

Example: To extract documents that contain US airport codes, you can create a concept named **`USAirports`** that includes these CLASSIFIER rules:

```
CLASSIFIER:BUF
CLASSIFIER:BUR
CLASSIFIER:BVK
```

So, documents that include a match on one or more of the airport codes **`BUF`**, **`BUR`**, or **`BVK`**, return a match for **`USAirports`**.

CONCEPT

Example: To extract documents that contain flight arrival information, create a concept named **`onTimeArrivals`**. The rule definition for **`onTimeArrivals`** contains the CONCEPT rule type. The CONCEPT rule type can reference the concept **`USAirports`** , which enables airport codes to be detected. The rule definition for the concept **`onTimeArrivals`** is as follows:

`CONCEPT:at USAirports on time`, where **`USAirports`** includes CLASSIFIER rules that identify US airport codes.

C_CONCEPT

Example: To extract documents that include names of university professors, create a C_CONCEPT rule named **`professorNames`** whose definition includes this rule:

`C_CONCEPT:Professor _c{firstName lastName}`

The rule indicates that matches are returned when **`firstName`** and **`lastName`** (previously defined) are found, but only when they are preceded by the word **`Professor`**. Provide the context for the match by using the modifier **`_c`** and enclosing the argument that you want to match in the braces (**`{}`**). The rule modifier **`_c{}`** indicates that the match occurs within the context of the specified concepts.

NO_BREAK

Example: Suppose you want to extract **`National Gallery of Art`**, but there also exists a concept named **`classTypes`** that includes the CLASSIFIER rule **`Art`**. You can create the following rule that prevents a partial match on **`classTypes`** and ensures that the entire string **`National Gallery of Art`** is matched:

`NO_BREAK:_c{USArtGalleries}`

The rule modifier **`_c`** indicates that the match occurs within the context of another concept.

REMOVE_ITEM

Example: Suppose you want to extract the baseball team St. Louis **`Cardinals`**, but not the football team Arizona **`Cardinals`**. You have a concept named **`football`** that includes the rule

CLASSIFIER:Cardinals. You have another concept named **baseball** that includes the rule CLASSIFIER:Cardinals. The following rule returns matches for the baseball team only:

```
REMOVE_ITEM(ALIGNED, "_c{football}", "baseball")
```

Note: The REMOVE_ITEM rule type could influence matches outside of the concept in which it is used. In this case, the rule could influence matches in the**football** rule because the rule specifies that items be removed.

REGEX

Example: To extract whole numbers in text (such as **1**, **23**, **456**, and so on), use the rule REGEX:[0-9]+. This rule requires that one or more consecutive digits occur and are without decimals.

Example: To extract a number that uses decimal notation, such as **392.55**, **45.25**, and **0,987654321**, use the following rule:

```
REGEX:[0-9]+[,\.][0-9]+
```

This rule returns a match on one or more digits, a comma, or a period, and then ending in one or more digits. For more information about writing Regex rules, see "Using Regular Expressions (Regex)" on page 97.

CONCEPT_RULE

Example: Suppose you want to extract Amazon the company, not Amazon the river. You could use this rule, which would return a company name within three words of the term **company**, but not if there were nature-related words in the document.

```
CONCEPT_RULE:(AND, (DIST_3, "_c{company}", "company"), (NOT, "natureTerms"))
```

SEQUENCE

Example: Suppose you want to extract first and last names only from a list of first, middle, and last names. You can use a SEQUENCE rule to define the arguments **first** and **last**. By using these arguments, matches are made on the concepts **firstName**, **middleName**, and **lastName**, but matches are returned on only **firstName** and **lastName**.

```
SEQUENCE:(first, last): _first{firstName} middleName _last{lastName}
```

PREDICATE_RULE

Example: Suppose you want to match a company to its products. You could use the following PREDICATE_RULE, which assumes that the concept **company** includes CLASSIFIER rules that list company names and the concept **products** contains CLASSIFIER rules that list products. Items must appear in the same sentence.

```
PREDICATE_RULE:(company, product):(SENT, "_company{copmany}","produces",
"_product{products}")
```

Writing Category Rules: Boolean Rules

Introduction to Category Rules

Category rules resolve to True or False. True results in a match. Category rules use Boolean and proximity operators, arguments, and modifiers to define the conditions that are necessary for category matches. Category rules are simpler to write than LITI rules and are recommended when there is no need to extract specific information from the data. For a list of Boolean and proximity operators, see "Boolean and Proximity Operators for Category Rules" on page 103.

Use the following syntax for a category rule:

(***OPERATOR***, *argument1*, *argument2*, ...)
where arguments can be terms, strings, or nested rules.

General rules for syntax:

- Boolean and proximity operators and their arguments are enclosed in parentheses and separated with commas. The arguments are included in quotation marks (“ ”). Example: `(AND, "my_w holiday", "_cap")`
- Rules can be nested. Example: `(AND, (OR, "courage", "courageous"), (OR, "brave", "bravery"))`
- Reference a category from another category by using special syntax called *tmac syntax* (_tmac). For more information, see “Using _tmac for Referencing Categories” on page 110.
- Concept names can be referenced in category rules. If you reference a concept, the concept matches are used to contribute to the true/false match of the category rule. Concept names must be enclosed in braces (**`[]`**). For example, to reference the concept **`gameShows`** in a category rule, you could write the rule `(OR, "[gameShows]").`

  Note: In categories, matches on concepts are based on an All Matches method, which returns all matches found in the text.
- Special symbols can be used to modify the rules to include, wildcards, case sensitivity, and so on. For a list of symbols, see “Using Symbols in Boolean Rules” on page 108.

Note: XPath expressions are not supported.

Boolean and Proximity Operators for Category Rules

The table below shows a list of Boolean and proximity operators that you can use to write category rules.

Table 12.10 Boolean and Proximity Operators for Category Rules

Operator	Description
AND	Takes one or more arguments. Matches if all arguments occur in the document, in any order. For example, the rule `(AND, "King", "Louis", "XIV")` returns a match if **`King`**, **`Louis`**, and **`XIV`** all occur in the document.
DIST_*n*	(Distance) Takes a value for *n* and two or more arguments. Matches if all arguments occur within *n* (or fewer) tokens of each other, regardless of their order. If an argument contains multiple tokens, then distance is calculated from the first token of the first argument to the first token of the last argument. **Note:** The `DIST_` operator does not use the same approach for calculating distance in a category rule that it does in a concept rule. For example, the rule `(DIST_5, "standard contract", "for the supply", "of goods")` returns a match in the phrase **`standard contract for the supply of goods`**. **Note:** For calculation purposes, the distance between tokens is not inclusive. For example, the distance between the tokens **`best`** and **`show`** in the phrase **`best in show`** is two tokens. Words that include hyphens are counted as one token (for example, **`merry-go-round`** is one token).

Operator	Description
END_*n*	(From the end of the document) Takes a value for *n* and one or more arguments. Matches if the argument occurs within *n* tokens from the end of the document. For example, the rule `(END_35, "conclusion")` returns a match if **`conclusion`** is found within 35 tokens from the last token in the document. Note: Words that include hyphens are counted as one word (for example, **`merry-go-round`** is one word).
MAXOC_*n*	(Maximum occurrence) Takes a value for *n* and one or more arguments. Matches if the document contains *n* or fewer occurrences of the arguments (in any order or combination). For example, the rule `(MAXOC_8, "savings", "offer", "best")` returns a match if **`savings`** occurs in the document six times. There is also a match if **`offer`** occurs in the document six times and **`best`** occurs twice.
MAXPAR_*n*	(Maximum paragraph) Takes a value for *n* and one or more arguments. Matches if all arguments occur within the first *n* (or fewer) paragraphs of the document, in any order. For example, the rule `(MAXPAR_4, "seasonal", "herbs", "plants")` returns a match if **`seasonal`** occurs in paragraph 4, **`herbs`** occurs in paragraph 2, and **`plants`** occurs in paragraph 2. Note: MAXPAR rules work properly only when applied to data sets that contain paragraph delimiters (\n\n). MAXPAR cannot be applied on the **Test Sample Text** tab. MAXPAR also cannot be applied in the **Categories** node to data that is contained in folders.
MAXSENT_*n*	(Maximum sentence) Takes a value for *n* and one or more arguments. Matches if all arguments occur within the first *n* sentences of the document, in any order. For example, the rule `(MAXSENT_4, "weight loss", "plan")` returns a match if **`weight loss`** and **`plan`** occur in sentence 3 of the document. For a list of sentence delimiters, see the SENT operator.
MIN_*n*	(Minimum) Takes a value for *n* and one or more arguments. Matches if the document contains at least *n* of the arguments specified (in any order). For example, the rule `(MIN_2, "Hollywood", "tinseltown", "movies")` returns a match if **`Hollywood`** and **`movies`** occur in the document. However, there is no match if **`Hollywood`** occurs twice and no other arguments occur.
MINOC_*n*	(Minimum occurrence) Takes a value for *n* and one or more arguments. Matches if the document contains at least *n* occurrences of the arguments specified (in any order or combination). For example, the rule `(MINOC_2, "Hollywood", "tinseltown", "movies")` returns a match if **`Hollywood`** and **`movies`** occur in the document. There is also a match if **`Hollywood`** occurs twice and no other arguments occur.

Operator	Description
NOT	Takes one argument. Matches if the argument does not occur in the document. Must be used with the AND operator. For example, the rule `(AND, (OR, "cinema", "theater", "theatre"), (NOT, "Broadway"))` returns a match if **cinema**, **theater**, or **theatre** occur in the document and **Broadway** does not. **Note:** The NOT operator applies across the entire document.
NOTIN	(Not in) Takes two arguments and matches if the first argument does not appear within the second argument. For example, the rule `(NOTIN, "butter", "peanut butter")` identifies **butter** when it does not appear within the noun phrase **peanut butter**. This sentence returns a match: **Early American colonists churned their own butter.**
NOTINDIST_*n*	(Not in distance) Takes a value for *n* and two arguments. Matches if the arguments do not occur within *n* tokens of each other, or if the first argument listed in the rule occurs in the document and the second argument does not. For example, the rule `(NOTINDIST_3, "orange", "green")` returns a match if **orange** and **green** do not occur within three tokens of each other, or if only **orange** appears in the document. The following sentence returns a match because the tokens that are specified in the rule are more than three words apart: **How green is my valley, how orange is the sunset?** **Note:** For calculation purposes, the distance between tokens is not inclusive. For example, the distance between the tokens **best** and **show** in the phrase **best in show** is two tokens. Tokens that include hyphens are counted as one token (for example, **merry-go-round** is one token).
NOTINPAR	(Not in paragraph) Takes two or more arguments and matches if all arguments occur within the document but appear in separate paragraphs. For example, the rule `(NOTINPAR, "China", "export")` returns a match if **China** and **export** occur in separate paragraphs (without the other argument present). **Note:** NOTINPAR rules work properly only when applied to data sets that contain paragraph delimiters (\n\n). NOTINPAR cannot be applied on the **Test Sample Text** tab. NOTINPAR also cannot be applied in the **Categories** node to data that is contained in folders.
NOTINSENT	(Not in sentence) Takes two or more arguments and matches when the first of the two arguments is present and the second of the two arguments does NOT occur. For example, the rule `(NOTINSENT, "trade", "China")` indicates that “trade” matches if the word “China” does not occur in the same sentence. For a list of sentence delimiters, see the SENT operator.

Operator	Description
OR	Takes one or more arguments. Matches if at least one argument occurs in the document. For example, the rule `(OR, "U.S.", "US", "United States")` returns a match if one or more of the items **`U.S.`**, **`US`**, or **`United States`** appear in the document. **Note:** Rules that are generated by SAS Visual Text Analytics nest the OR operator within the AND operator. However, the OR operator can stand alone.
ORD	(Order) Takes one or more arguments. Matches if all of the arguments occur in the order that is specified in the rule. It cannot be used with SENT (or any other operator that limits the scope of matches). For example, the rule `(ORD, "warranty", "claim", "denied")` returns a match in the sentence **`The warranty claim for the washing machine was denied.`**
ORDDIST_*n*	(Order and distance) Takes a value for *n* and two or more arguments. Matches if both arguments occur in the same order that is specified in the rule and if both arguments are within *n* tokens of each other. If an argument contains multiple tokens, then distance is calculated from the last token of the first argument to the first token of the last argument. **Note:** The `ORDDIST` operator does not use the same approach for calculating distance in a category rule that it does in a concept rule. For example, the rule `(ORDDIST_4, "standard contract", "for the supply", "of goods")` returns a match in the phrase **`standard contract for the supply of goods`**. **Note:** For calculation purposes, the distance between tokens is not inclusive. For example, the distance between the tokens **`best`** and **`show`** in the phrase **`best in show`** is two tokens. Words that include hyphens are counted as one token (for example, **`merry-go-round`** is one word).
PAR	(Paragraph) Takes one or more arguments. Matches if all the arguments occur in a single paragraph, in any order. For example, the rule `(PAR, "director", "budget")` returns a match if the paragraph includes both **`director`** and **`budget`**. **Note:** PAR rules work properly only when applied to data sets that contain paragraph delimiters (\n\n). PAR cannot be applied on the **Test Sample Text** tab. PAR also cannot be applied in the **Categories** node to data that is contained in folders.

Operator	Description
PARPOS_*n*	(Paragraph position) Takes a value for *n* and one or more arguments. Matches if all arguments occur within the n^{th} paragraph, in any order. For example, the rule `(PARPOS_2, "journalists", "detained", "overseas")` returns a match if **`journalists`**, **`detained`**, and **`overseas`** occur within paragraph 2 of the document. **Note:** PARPOS rules work properly only when applied to data sets that contain paragraph delimiters (\n\n). PARPOS cannot be applied on the **Test Sample Text** tab. PARPOS also cannot be applied in the **Categories** node to data that is contained in folders.
SENT	(Sentence) Takes two or more arguments. Matches if all the arguments occur in the same sentence, in any order. For example, the rule `(SENT, "growth", "hormone")` returns a match in the sentence **`Patients who take a growth hormone might experience side effects`**. For a list of sentence delimiters that can be used with the SENT operator, see Table 12.11 on page 107.
START_*n*	(From the start of the document) Takes a value for *n* and one or more arguments. Matches if the argument occurs within *n* tokens from the start of the document. For example, the rule `(START_22, "infection")` returns a match if **`infection`** occurs within 22 tokens of the first word in the document. **Note:** Words that include hyphens are counted as one token (for example, **`merry-go-round`** is one token).

Table 12.11 *Sentence Delimiters for the SENT Operator*

Delimiter	Description
`\r\n\r\n`	Two consecutive carriage returns and new lines (for documents created in Windows)
`\r\n \r\n`	Two consecutive carriage returns and new lines, separated by a space
`.<SPACE>`	Period (.) followed by an ASCII space
`.\n`	Period (.) followed by a new line
`.\r`	Period (.) followed by a carriage return
`!`	Exclamation point
`!\n`	Exclamation point followed by a new line
`!\r`	Exclamation point followed by a carriage return

Delimiter	Description
`?`	Question mark
`?\n`	Question mark followed by a new line
`?\r`	Question mark followed by a carriage return
`.)`	Period followed by a closing parenthesis
`!)`	Exclamation point followed by a closing parenthesis
`?)`	Question mark followed by a closing parenthesis
`.”`	Period followed by double quotation marks

Using Symbols in Boolean Rules

To modify your Boolean rules for category matching, you can use the symbols listed in “Using Symbols in Boolean Rules” on page 108. Symbols are written as suffixes to strings in arguments. For example, to expand the word **`breathe`** to all inflected verb forms, which include **`breathes`** and **`breathing`**, use the following syntax for the argument: **`“breathe@V”`**.

Table 12.12 Special Symbols Used in Boolean Rules

Symbol	Description
*	(Wildcard matching) Matches any characters that occur at the beginning or end of the word. For example, the argument **`“travel*”`** returns the matches **`travels`** , **`traveled`**, **`traveler`**, **`traveling`**, and so on. The argument **`“*room”`** matches **`bedroom`**, **`cloakroom`**, **`ballroom`**, **`room`**, and so on.
^	Beginning of sentence) Starts searching at the beginning of the sentence to find a match. For example, the argument **`“^Independent”`** returns a match in this sentence: **`Independent research was conducted.`** Note: Tokens (words, phrases, symbols, or other meaningful elements) need to be entered specifically to be considered for matching. For example, if you are searching for **`**In this case`**, use the argument **`“^\*\*In this case”`**. Also note that backward slashes (\) are used as escape characters for the asterisks (*) so that the asterisks are not treated as wildcards.

Symbol	Description
$	(End of sentence) Starts searching at the end of the sentence to find a match. For example, the argument **`"deleted.$"`** returns a match on the following sentence: **`All the files were hastily deleted.`** **Note:** Tokens (words, phrases, symbols, or other meaningful elements) need to be entered specifically to be considered for matching. For example, the argument **`"deleted$"`** would not produce a match on the following sentence: **`All the files were hastily deleted.`** because the ending period (.) was not specified.
@	(Morphological expansion) Expands the category rule to match all inflectional forms of the word in the argument. For example, the argument **`"wonder@"`** returns the matches **`wonder`**, **`wonders`**, **`wondered`**, **`wondering`**, and so on (but does not return a match on **`wonderful`**). **Note:** If you apply @ to a word that SAS Visual Text Analytics does not recognize, no expansion occurs. Only the exact string specified before the @ is returned. For example, **`"grath"`** would not expand. Only the string **`grath`** would return a match in the rule.
@A	(Morphological expansion for adjectives) Expands the category rule to match inflected comparative and superlative adjective forms of the word in the argument. For example, the argument **`"happy@A"`** returns the matches **`happier`** and **`happiest`**. **Note:** If you apply @A to a word that is not an adjective, no expansion occurs.
@N	(Morphological expansion for nouns) Expands the category rule to match all noun forms of the word in the argument. For example, the argument **`"quality@N"`** returns the matches **`quality`** and **`qualities`**. **Note:** If you apply @N to a word that is not a noun, no expansion occurs.
@V	(Morphological expansion for verbs) Expands the category rule to match all verb forms of the word in the argument. For example, the argument **`"transfer@V"`** returns the matches **`transfer`** , **`transfers`**, **`transferred`**, and **`transferring`**. **Note:** If you apply @V to a word that is not a verb, no expansion occurs.
_L	(Literal matching) Matches a literal string. Useful when you want to match a string that includes symbols. For example, the argument **`"$USD_L"`** returns the match **`$USD`**. **Note:** Tokens (words, phrases, symbols, or other meaningful elements) need to be entered specifically to be considered for matching.

Symbol	Description
_C	(Case matching) Specifies case-sensitive matching. For example, the argument **"Iris_C"** returns the match **Iris**, but not **iris**.

Using _tmac for Referencing Categories

Referencing a category enables you to use the rules in an existing category without having to duplicate the rules. Use tmac syntax (_tmac) to reference an existing category in a category rule. The definition of the existing rule is processed in the category that references it.

To reference a category, you must identify its path. All category paths begin with **@**. From there, you can specify the path by following the category hierarchy.

For example, suppose you have the following category structure under **All Categories**:

NAME
    FIRST

    LAST

You would reference the category **FIRST** as **@NAME/FIRST**.

You can use the tmac syntax with Boolean operators. For example, suppose you want to reference the category **FIRST** from a category called **FIRST_NAME**. You could add this rule in the **FIRST_NAME** definition:

```
(OR,_tmac:"@NAME/FIRST")
```

To enforce a first name followed by last name (FIRST LAST), you could add this rule in a category called **COMPLETE_NAME**:

```
(ORD,_tmac:"@NAME/FIRST",_tmac:"@NAME/LAST")
```

The definitions written in **FIRST** and **LAST** are automatically processed.

13

Recommended Reading

Recommended Reading

Here is the recommended reading list for this title:

- *SAS Text Analytics for Business Applications: Concept Rules for Information Extraction Models*
- *Text Mining and Analysis: Practical Methods, Examples, and Case Studies Using SAS*

Appendix 1

Part-of-Speech Tags (for Languages Other Than English)

Introduction to Part-of-Speech and Other Tags

The part-of-speech tags for rule writing for languages other than English are listed in the following tables. Also included are other tags that are not considered parts of speech (such as punctuation). All tags are case-sensitive and are preceded by a colon (:) in concept rules. For more information, including English tags, see "Using Part-of-Speech and Other Tags" on page 95.

Part-of-Speech Tags for Rule Writing

Arabic

Table A1.1 Part-of-Speech Tags for Arabic

Part-of-Speech Tag	Description	Examples
:ADJ	Adjective	أبدي, أثري
:ADV	Adverb	أيضا, ربما
:CONJ	Conjunction	بل, حتى
:DET	Determiner	اال
:DIALECT	Dialect	آسم, أثول
:FUT	Future particle	س, سوف
:INTERJ	Interjection	أجل, لا
:INTERROG	Interrogative	أين, عمّا
:NEGPART	Negative particle	لم
:NOUN	Noun	تفاحة, شجرة
:NUM	Number	آلاف, أربعة
:PART	Particle	قد, لقد
:PREP	Preposition	إلاّ, على
:PRON	Pronoun	أنا, أنت
:PROP	Proper noun	أمريكا
:PUNC	Punctuation	؟, ،

Part-of-Speech Tag	Description	Examples
:CV	Imperative verb	ائتيا, العبان
:IV	Present verb	تأتون, تلعبا
:PV	Past verb	أتتا, لعبت
:ASCII	English word	memory, tablets
:DEFAULT	Unknown word	اعتياديًا, وشيئٌ
:NUMBER	Number	1.8, 200
:URL	URL	www.sas.com

Chinese

Table A1.2 Part-of-Speech Tags for Chinese

Part-of-Speech Tag	Description	Examples
:A	Adjective	俊俏, 开心, 兇險, 凌亂
:ASCII	ASCII characters in half-width and full-width	sas, do, happy, day2136456, ＡＰＥＣ，Ｇ２０
:C	Conjunction	或, 与, 雖然
:D	Adverb	非常, 偏偏, 稍微, 永遠
:digit	Number	1051, 1.9
:E	Interjection	咦, 呸, 哦哟
:F	Location/direction	中間, 下边, 南侧
:G	Other morpheme	馨, 慚
:H	Other prefix	亚, 非
:K	Other suffix	们, 者, 們
:L	Idiom (chengyu)	囫囵吞枣, 博古通今, 一廂情願
:M	Quantifier	十, 卅, 成千上万, 上萬, １０５１
:N	Noun	人, 桌子, 香蕉, 枷鎖
:NR	Proper noun, name	习近平, 梁振英, 奥巴马

Part-of-Speech Tag	Description	Examples
:NR_xing	Proper noun, last name for Chinese (most are single characters)	赵, 邹, 诸葛, 趙
:NS	Proper noun, geographic	中国, 美國, 山東
:NS_abbr	Proper noun, abbreviation for country names (all are single characters)	俄, 匈, 葡, 緬
:NT	Proper noun, organization	北京大学, 上汽集團
:NZ	Proper noun, miscellaneous	潘婷, 劍南春
:O	Onomatopoeia	吱呀, 叽叽喳喳, 劈裏啪啦
:P	Preposition	依照, 对于
:Punct	Punctuations or symbols (the majority are English)	, ! ? % @ ($
:sep	Separator (English period)	.
:Q	Classifier	个, 斤, 艘, 加侖
:R	Pronoun	我, 他們, 这
:S	Subcountry location (general; specifics only within sinosphere)	地上, 上空, 高处, 內廳
:T	Temporal phrase	今天, 夜间, 十月, 去歲
:U	Particle	的, 了, 着
:UNKNOWN	Unknown word	婳, 縪
:inc	Unknown word	妍
:V	Verb	看, 认为, 彈奏, 徵納
:W	Punctuation or symbols	。，！？%@ $
:Y	Interjectional particle	吧, 吗, 麽
:date	Date (Only ISO week date)	2003–W52–6, ２００３－Ｗ５２－６
:time	Time	23:59:59, 2000-01-01T00:00:00, ２０：１６：２０，２００８／５／２６／１１：５４
:url	URL, pathname, and email address	www.sas.com

Croatian

Table A1.3 *Part-of-Speech Tags for Croatian*

Part-of-Speech Tag	Description	Examples
:A	Adjective	svaki, hrvatskim, koje
:ADV	Adverb	uistinu, tamo
:CONJ	Conjunction	a, ali, kad
:INTJ	Interjection	hej, hajde, oh
:N	Noun	dan, april, dr, itd.
:PTCL	Particle	ne, bilo (as in “bilo koje”)
:PPOS	Preposition	sa, bez, o
:PRO	Pronoun	ja, me, ih, nas, vam, njihovoj, svašta
:V	Verb	voli, došao, pozvala, dođite, bih
:NUM	Number	2, dva, sedmi, 1.23.2015
:time	Time	23:30:01
:PUNC	Separator or punctuation	, .
:PN	Proper noun	Aleksandar, Jelenu, Gorenje, Zagreb

Czech

Table A1.4 *Part-of-Speech Tags for Czech*

Part-of-Speech Tag	Description	Examples
:A	Adjective	duchovní, celý, všechny, čertvíjaký, která, jakém, žádnej
:ADV	Adverb	například, dál, zároveň, někam, ne
:CONJ	Conjunction	a, nebo
:INTJ	Interjection	ahoj, fuj
:N	Noun	autorů, lidem
:NUM	Spelled out number	tři, dvoje, šestatřicáté

Part-of-Speech Tag	Description	Examples
:digit	Number	33, 1844, 14.3.2014
:PPOS	Preposition	v, z
:PRO	Pronoun	kdo, sobě, nás, tomto, tím, nikoho, nic, její, mou
:V	Verb	nebyl, jdou
:sep	Separator or punctuation	. , :
:PN	Proper noun	Pavel, Valenta, Chotěbořským
:inc	Unknown or foreign word	mp3, larger
:time	Time	23:30:01
:url	URL	www.sas.com, http://www.sas.com

Danish

Table A1.5 Part-of-Speech Tags for Danish

Part-of-Speech Tag	Description	Examples
:A	Adjective	socialest, udartendere
:ADV	Adverb	sydsydøst
:CONJ	Conjunction	Såsom
:DET	Number	den
:INTJ	Interjection	joh, pøj
:N	Noun	thyboernes, centerer, DVS, FL, ibm, netscape, tirsdag
:NUM	Number	tyvefem, tredive
:PN	Proper noun	Egholm, Franck, Carlos, Mallorca, Groth, Leth, Renault, Corel
:PPOS	Preposition	fra, trods
:PRO	Pronoun	dens, hans, jerselv, sigselv
:V	Verb	opofre, læsende, anvender, bliver, tredjebehandlet, læste, læse, tilvirk, bemyndiges, fuldkommengøredes

Part-of-Speech Tag	Description	Examples
:date	Date	23-12-2012, 12/12/2012
:time	Time	:23:50, 09:23
:digit	Digit	2012, 12.23
:url	Internet address	http://www.sas.com
:sep	Separator or punctuation	. , ;
:inc	Unknown word	bl, erne

Dutch

Table A1.6 *Part-of-Speech Tags for Dutch*

Part-of-Speech Tag	Description	Examples
:A	Adjective	betrouwbaar, gelukkig, mooi
:ADV	Adverb	eenmaal, hier, nu
:CONJ	Conjunction	als, doch, hoe
:DET	Determiner	de, der, een, ten, ter
:digit	Number	21
:NUM	Numeral	acht, elf, miljard, duizend
:inc	Unknown word	xrxx
:N	Noun	geluk , schoonheid, kg, zgn, anti, hoofde, tijde, voordele
:PN	Proper noun	Amerika, Nederland
:PPOS	Preposition	met, per, te, van
:PRO	Pronoun	alles, beide, hetgeen
:sep	Separator or punctuation	,
:url	URL	http://www.sas.com
:V	Verb	helpt, vernieuwt, helpen, vernieuwen, helpende, vernieuwende, geholpen, vernieuwd

English

Table A1.7 Part-of-Speech Tags for English

Part-of-Speech Tag	Description	Examples
:A	Adjective	luckier, worse, mellowest, merriest
:ADV	Adverb	lyrically, physically, luckier, worse
:CONJ	Conjunction	when, yet, how, when, whereby
:date	Date	04/03/2012
:digit	Sequence of Numbers	2345, 234.22, 21/234
:DET	Determiner	the, an, every, our, his, my, such, all
:inc	Unknown word	slaster, lijer
:INTJ	Interjection	hah, hello
:N	Noun	love, sheep, shoes, etc., Ms, cm, facto, klieg, modus
:NUM	Number	twenty, hundred
:PN	Proper noun	SAS, Cary, Goodnight
:PPOS	Preposition	on, under, across, after, except, away, forward, in, ex, multi
:PRO	Pronoun	he, one, somebody, me, myself, oneself, yours, hers, which, whatever, whose, whoever
:sep	Separator or punctuation	; , /
:time	Time	7AM, 10:00
:url	Filenames, pathnames, URL	A:/mydir/file.txt, www.sas.com
:V	Verb	be, do, have, am, can, should, will, goes, sees, is, does, doing, having, climbing, been, had, was, were, did, have, dashed, factored, went

Farsi

Table A1.8 *Part-of-Speech Tags for Farsi*

Part-of-Speech Tag	Description	Examples
:A	Adjective	خوشگل, خوشحال
:Acomp	Comparative Adjective	خوشگل‌تر, خوشحال‌تر
:Asup	Superlative adjective	خوشگل‌ترین, خوشحال‌ترین
:Appl	Participle used as adjective	آسایانیده, آبانانده
:ADV	Adverb	هنوز, آنگه, ابتدائا:
:CLASS	Classifier	باب, تخته, رأس
:CONJ	Conjunction	اگر, تااینکه
:DET	Determiner	اون, این
:INTJ	Interjection	آه, آفرین, ای
:N	Noun	آذوقه, آرنج, چشم
:Npl	Plural noun	آرنج‌ها, چشم‌ها
:NUM	Numeral	دو, صد, میلیون
:NUMord	Ordinal numeral	دومین, سوم, صدمین
:PN	Proper noun	اسرائیل, آتوسا
:PPOS	Preposition	از, الا, چون
:PRO	Pronoun	ن, او, شما
:PUNC	Punctuation or symbol	“ (؟ %
:Vinf	Infinitive (usage similar to English gerund)	خواندن ,خوردن
:V	Verb	بخوان, بخوانم, خواندم
:ASCII	ASCII characters and digits	happy, 2017, love123
:DEFAULT	Unknown word	بخوانبخوان

Finnish

Table A1.9 Part-of-Speech Tags for Finnish

Part-of-Speech Tag	Description	Examples
:A	Adjective	loistava, korkea
:ADV	Adverb	ohitse, juuri
:CONJ	Conjunction	ja, vaan, ellej, jotta
:date	Date	2001-12-02
:digit	Number	1234, 7
:inc	Unknown word	auttonkkan, eggs
:N	Noun	siltoineen, postiksi
:PN	Proper noun	Pertti, Fazer
:PPOS	Preposition	pitkin, kanssaan
:PRO	Pronoun	noihin, muussa, ketkä
:sep	Separator or punctuation	; / +
:time	Time	12:00:00, 7PM
:url	URL	http://www.sas.com
:V	Verb	heilahtamassa, heilauttaen, olla, kinko, pas, lähennemme, kumarrettava, jaettu, meditoitpa, ihastele, omistautuisi, pakkaa

French

Table A1.10 Part-of-Speech Tags for French

Part-of-Speech Tag	Description	Examples
:A	Adjective	comparable, compassionnelle, intraduisibles
:ADV	Adverb	plutôt, individuellement
:CONJ	Conjunction	et, ou, lorsque, puisque

Part-of-Speech Tag	Description	Examples
:DET	Determiner	sa, tes, ce
:digit	Number	123, 12.3, 12.3.2003, 12/3/2003
:inc	Unknown word	analytics
:INTJ	Interjection	tralala, zzz
:N	Noun	zèbre, encyclopédie
:PN	Proper noun	Eurotunnel, Égypte
:AFX	Affix	anglo, éco
:PPOS	Preposition	jusque, aux, du
:PTCL	Particle	vitae, ab
:PRO	Pronoun	lui
:sep	Separator or punctuation	, . !
:url	URL	http://www.sas.com
:V	Verb	vais, obligez, travaillées, traduire, tramant

German

Table A1.11 *Part-of-Speech Tags for German*

Part-of-Speech Tag	Description	Examples
:A	Adjective	zuverlässig
:ADV	Adverb	gern, sehr
:CONJ	Conjunction	und, oder
:DET	Determiner	eine, manch
:digit	Number	21
:NUM	Numeral	fünf, zwölf
:EMP	Emphatic or intensifier	ganz
:inc	Unknown word	xrxx
:N	Noun	Schönheit, Zuverlässigkeit

Part-of-Speech Tag	Description	Examples
:PN	Proper noun	Mozart, Nirvanas, Niederlanden
:PPOS	Preposition	kontra, ober, lob
:PRO	Pronoun	er, sie, der, heraus
:sep	Separator or punctuation	,
:url	URL	http://www.sas.com
:V	Verb	ging, half, gehen, helfen

Greek

Table A1.12 Part-of-Speech Tags for Greek

Part-of-Speech Tag	Description	Examples
:A	Adjective	ενορμητικός, άβαθος
:ADV	Adverb	πολύ, επίσης
:CONJ	Conjunction	και, αλλά
:DET	Determiner	ένας, ο
:INTJ	Interjection	χαίρε, όπα
:N	Noun	μήλο, δέντρο
:PTCL	Particle	πάρα
:PPOS	Preposition	άχρι, διά
:PRO	Pronoun	εσύ, αυτός
:V	Verb	παίσαμε, παίνεψε, παίξει, παίζαμε, παίζουμε, παίζοντας, παίρνοντάς, κατασκευαστώ, έλα
:url	URL	http://www.sas.com
:date	Date	2015-12
:digit	Number	1, 20
:sep	Separator or punctuation	. , »
:inc	Unknown word	χλμ

Part-of-Speech Tag	Description	Examples
:time	Time	23:59
:PN	Proper noun	Μάντσεστερ

Hebrew

Table A1.13 *Part-of-Speech Tags for Hebrew*

Part-of-Speech Tag	Description	Examples
:A	Adjective	יפה, אדיר
:ADV	Adverb	באמת, בבטחה
:CONJ	Conjunction	או, בגלל
:INTJ	Interjection	אוף, אהה
:N	Noun	רחוב, ברחוב, אבזור, אבטחה
:PN	Proper noun	ישראל, אבוג'ה, אדוארד
:PPOS	Preposition	אודות, אצל
:PRO	Pronoun	אנחנו, באתה, ה"הן, מהיכן
:NUM	Quantifier	אחד, ביליון, שתיהן
:V	Verb	שמח, אבטח, אהבו
:date	Date	12/31/2016, 2016-12-31
:digit	Number	100, 6,666, 6.000
:inc	Unknown word	happy, happy123, בוויטנאם
:sep	Separator or punctuation	. , ! -
:time	Time	14:30:30
:url	URL	http://www.sas.com

Hindi

Table A1.14 Part-of-Speech Tags for Hindi

Part-of-Speech Tag	Description	Examples
:A	Adjective	ज्ञात, ज्ञानी
:PRO	Pronoun	तेरा, मेरा
:N	Noun	मेयर, मैग्नोलिया
:ADV	Adverb	यथायोग्य, यथोचित
:CONJ	Conjunction	यदि, यद्यपि
:DET	Determiner	ऐसा, इसी
:INTJ	Interjection	आह, अहा
:NUM	Number	अस्सी, अड़तालीस
:PN	Proper noun	अग्नीवो
:PPOS	Particles	का, का
:V	Verb	खरीदना, गुजर
:PUNC	Separator or punctuation	।, ॥
:sep	Separator or punctuation	,.)
:inc	Unknown word	आिद, २२५
:digit	Number	0, 3

Hungarian

Table A1.15 Part-of-Speech Tags for Hungarian

Part-of-Speech Tag	Description	Examples
:A	Adjective	természetes, gyors
:ADV	Adverb	néha, gyorsan
:AFX	Affix	meg, el
:CONJ	Conjunction	és, de

Part-of-Speech Tag	Description	Examples
:DET	Determiner	a, az, egy
:INTJ	Interjection	köszi, no
:N	Noun	ablakoknak, zsiráfra
:NUM	Spelled out number	tízezer, száznegyven
:PN	Proper noun	Angliában, Andrea
:PPOS	Pre- or Postposition	szerint, alatt
:PRO	Pronoun	annak, velem
:V	Verb	utazgatok, vagyunk
:date	Date	2003.12.18., 25-én
:digit	Digit	16, 2014
:inc	Unknown word	inconnue
:sep	Separator or punctuation	!?%$
:time	Time expression	22:40
:url	URL	http://metin2univers.mindenkilapja.hu

Indonesian

Table A1.16 *Part-of-Speech Tags for Indonesian*

Part-of-Speech Tag	Description	Examples
:A	Adjective	lonjong, menjengkelkan
:N	Noun	kosmologiku, lotengnya, dpa
:ADV	Adverb	mingguan, perlahan
:CONJ	Conjunction	sambil, biarpun
:V	Verb	biaskanlah, membuntutiku
:DET	Determiners	sebuah
:NUM	Number words	empat, delapan
:INTJ	Interjections	hai, hoi

Part-of-Speech Tag	Description	Examples
:PRO	Pronoun	dikau, engkau
:PN	Proper noun	irlandia, filipina
:PPOS	Phrasal; the word can be combined with another word to form a phrase	sebiru, secantik
:sep	Separator or punctuation	"(,
:inc	Unknown words	jpg, png
:digit	Number	22, 490
:url	URL	www.jakarta.go.id
:date	Date	12/31/2016

Italian

Table A1.17 Part-of-Speech Tags for Italian

Part-of-Speech Tag	Description	Examples
:A	Adjective	affidabile, bellissimo, felice, felicemente, rapidamente
:CONJ	Conjunction	ma, oppure, sebbene
:DET	Determiner	il, la, uno
:digit	Number	21
:inc	Unknown word	ah, ahimè
:INTJ	Interjection	Xrxx
:N	Noun	affidabilità, bellezza, felicità
:PN	Proper noun	Roma, Italia
:PRO	Pronoun	io, ne
:PPOS	Preposition	con, in, per, anti, ri, anza, issimo
:sep	Separator or punctuation	,
:url	URL	http://www.sas.com
:V	Verb	andare, andando, andasse, andato

Japanese

Table A1.18 Part-of-Speech Tags for Japanese

Part-of-Speech Tag	Description	Examples
:AJ	Adjective	長い, 忙しい,便利だ
:AV	Adverb	いかが, やはり
:AVC	Adverbs of form or condition	直に, ぐっすり
:AVD	Adverb of degree	とっても, 大して
:AVE	Adverb of evaluation	たまたま, 無論
:AVF	Adverb of frequency	あくまで, しばしば
:AVO	Adverb of opinion	いわば, 概して
:AVQ	Adverb of quantity	大方, いくら
:AVS	Adverb of statement	いかに, あたかも
:AVT	Adverb of tense or aspect	急遽, 直ぐ
:AX	Auxiliary verbs	べきだ, らしい, ようだ
:CN	Conjunction	並びに, 但し, だけど
:CP	Copula	だ, なんだ
:DA	Adverbial demonstrative	こう, そう, あのように
:DM	Prenominal demonstrative	この, あの, そん な
:DN	Pronoun	あれ, こちら, あそこ
:MD	Prenominal modifier	小さな, 主たる, 色ん な
:IT	Interjection	あれれ, あ～, え えと
:NA	Adverbial noun	おおむね, なにぶん
:NC	Common noun	風, 学校, 雑誌
:NK	Content noun	の, もの, こと
:NT	Noun of time	長年, 夏, 先月
:NV	Verbal noun	請求, 弁解, 勉強
:NP	Proper noun	ＷＴＯ繊維協定, 米州

Part-of-Speech Tag	Description	Examples
:NH	Proper noun of Person	中川秀直, 中川浩明, 中川勝
:NHM	Proper noun of Given name	奈江子, 太郎, 那恵子
:NHS	Proper noun of Family name	鈴木, 佐藤, 田中
:NPO	Proper noun of Organization	米軍, 米国, 米国際貿易委員会
:NL	Proper noun of Place	米国, 越南, 奈央島
:NN	Numeral	千, 零, 6
:PC	Particles of case marker	を, で, の, へ
:PE	Particles that appear at the end of the sentence	っけ, な, なぁ
:PN	Particles that combine nominals	ないし, ないしは, 並びに
:PP	Particles that combine clauses	ながら, なら, のに
:PQ	Particles of quotation	て, と, っと
:PS	Particles that mean *only* or *too*	も, のみ, くらい
:PRJ1	Prefixes to i-adjective	か, こ, 真
:PRJ2	Prefixes to na-adjective	無, 不, 非
:PRN	Prefixes to nominals	高, 前, 全
:PRV	Prefixes to predicates	相, 猛, 最
:SJN	Suffixes to nouns and configure adjectives	っぽい, くさい
:SJV	Suffixes to verbs and configure adjectives	たい, づらい
:SNA	Suffixes to adjectives and configure nouns	さ
:SNC	Suffixes to classifiers and configure nouns	せんち, ぺーじ
:SNN	Suffixes to nouns	っ子, 中, 所
:SNV	Suffixes to verbs and configure nouns	かた, っぷり
:SV	Suffixes to verbs	せる, れる, 上げる
:V1	Ichidan Verb	治せる, 泣ける, 叫べる

Part-of-Speech Tag	Description	Examples
:V5	Godan Verb	直す, 長びく, 産む
:VK	Kuru Verb	来る
:VS1	Suru Verb	する
:VS2	Suru Verb d	賀する, 刑する, 御する
:VSN	Suru Verb	きりきり, 毅然と
:VZ	Zuru verb	準ずる, 同ずる
:SC	Special category-comma	、 ，
:SCP	Special category-closed parentheses	）》 ］
:SOP	Special category-opened parentheses	（ 《［
:SK	Special category-other symbols	？… ～
:SP	Special category-period	。 ．
:SS	Special category-space	
:digit	Number	1.0, 10
:sep	Separator or punctuation	. ,
:KATAKANA	Unknown word in katakana	ポータブルオプション, オブザベーション
:HIRAGANA	Unknown word in hirakana	きんぽうげ
:UNKNOWN	Unknown word	嘘, 甦
:ASCII	English word	Display, Ｍｏｍｅｎｔｅ

To use Japanese POS tags in LITI rules, you need to add the Form type after the POS tags. For the POS tags of nominals, add '|ROOT' after the POS tags, for example, 'NC|ROOT', 'DN|ROOT', 'CN|ROOT'. For the POS tags of predicates, add the conjugation forms listed in the table below, for example, 'AJ|CONJ', 'V1|COND'.

Form Type	Japanese Description	English Description	Examples
ROOT	体言基本形	Basic form of nominals	お花, 手
BS	用言基本形	basic form of predicates	読む, 速い
BSDEA	デアル列基本形	dearu basic conjunctive	静かである
BSWR	デス列基本形	desu basic	静かです
COND	文語基本形	written basic form	あいさつす

Form Type	Japanese Description	English Description	Examples
CONDDEA	デアル列条件形	basic euphony conditional	読めば, 読みや, 速ければ, 速けりや
CONDDEATA	デアル列タ系条件形	dearu ta conditional	静かであれば
CONDDESTA	デス列タ系条件形	desu ta conditional	静かであったら
CONDTA	タ系条件形	ta conditional	静かでしたら
CONDWR	文語条件形	written conditional	読んだら, 速かったら
CONJ	基本連用形	basic conjuctive	読め
CONJDEA	デアル列基本連用形	dearu conjuctive-tari form	読み (ます) , 速く, 静かに
CONJDEATA	デアル列タ系連用テ形	dearu/ta conjunctive-te form	静かであり
CONJDEATARI	デアル列タ系連用タリ形	dearu ta conjunctive-tari form	静かであったり
CONJDESTARI	デス列タ系連用タリ形	desu ta conjunctive-tari form	静かでしたり
CONJDESTE	デス列タ系連用テ形	desu ta conjunctive -te form	静かでして
CONJTARI	タ系連用タリ形	ta conjunctive -tari form	書いたり, 速かったり
CONJTE	タ系連用テ形	ta conjunctive -te form	書いて, 速くて
CONJWR	文語連用形	written conjunctive	あいなう, あかう
DEATA	デアル列タ形	dearu ta form (plain past tense)	静かであった
DESTA	デス列タ形	desu ta form	静かでした
IMP	命令形	imperative	読め, 速かれ, 静からレ
IMPDEA	デアル列命令形	dearu imperative	であれ, 静かであれ
IMPWR	文語命令形	written imperative	あいさつせよ
INT	意志形	intention form	読もう
IPE	未然形	Imperfective	読ま (ない)
IPEDEAWR	デアル列文語未然形	written -dearu imperfective	べきであら
IPEWR	文語未然形	written imperfective	速から (ず)
KANO	可能形	form that attaches to can words	太れ, 失え

Form Type	Japanese Description	English Description	Examples
PASS	受身形	form that attaches to passive forms	失わ
PERF	完了形	form that attaches to perfective	失効し
PNOM	ダ列基本連体形	basic prenominal	速き (こと), 静かな, 上等の
PNOMWR	文語連体形	written prenominal	失き, 好きずききき
PSU	基本推量形	(-da) basic presumptive	速かろう, 静かだろう
PSUDEA	デアル列基本推量形	dearu presumptive	好きであろう
PSUDEATA	デアル列タ系推量形	dearu ta presumptive	静かであったろう, であったろう
PSUDES	デス列基本推量形	desu presumptive	好きでしょう
PSUDESTA	デス列タ系推量形	desu ta presumptive	好きでしたろう
PSUTA	タ系推量形	ta presumptive	読んだろう, 速かったろう, 静かだったら
SHIEKI	使役形	form that attaches to causatives	あいさつさ
TA	タ形	ta form (plain past tense)	読んだ, 速かった, 静かだった

Korean

Table A1.19 Part-of-Speech Tags for Korean

Part-of-Speech Tag	Description	Examples
:AD	Adverb	매우, 정말, 빨리
:AJ	Adjective	예쁘다, 귀엽다, 차분하다
:GAC	Case grammatical affix	가, 를, 로
:GAD	Determinative grammatical affix	은, 을, 는
:GAH	Change grammatical affix	이다, 기, 음
:GAJ	Conjunctive grammatical affix	는데, 는지, 느라고
:GAP	Predicate grammatical affix	다, 습니다, 더구만
:GAR	Respect grammatical affix	시, 으시, 옵

Part-of-Speech Tag	Description	Examples
:GAT	Time grammatical affix	겠, 었, 였었
:GAX	Auxiliary grammatical affix	도, 만, 까지
:IJ	Interjection	아, 네, 그래
:NN	Noun	하늘, 산, 바다
:NNB	Bound noun	것, 수, 개
:NNP	Proper noun	서울, 이순신, 국립국어원
:NUMBER	Number	하나, 둘, 셋
:PF	Prefix	제-, 햇-, 명-
:PN	Prenoun	각, 첫,기초적
:PR	Pronoun	이것, 언제, 이분
:PUNC	Punctuation	. ? ! ( )
:SF	Suffix	-꾼, 꾸러기, -감
:VB	Verb	웃다, 뛰다, 날다
:ASCII	English Word	Korean, iPhone, SK
:DATE	Date	2015-04-28, 20150428
:DEFAULT	Unknown word	하페즈, 샤리프, 쿠레쉬
:TIME	Time	23:59:59
:URL	URL	http://www.sas.com

Norwegian

Table A1.20 Part-of-Speech Tags for Norwegian

Part-of-Speech Tag	Description	Examples
:A	Adjective	leket
:ADV	Adverb	alltid, framover
:CONJ	Conjunction	som
:date	Date	12/23/2012, 23/12/2012

Part-of-Speech Tag	Description	Examples
:DET	Preposition+determiner	idette, idenne
:INTJ	Interjection	hm
:N	Noun	anordningen, tydeets, mfl, mht, tusen, seks, sms
:PN	Proper noun	Egholm, Puccini, Tertnes, Høyem, Lundberg, Braathens, ruskursus, ørknen
:NUM	Number	12, 23, 23.4
:PPOS	Preposition	fra
:PRO	Pronoun	jeg, det, dens, sjølve
:PUNC	Punctuation	, . !
:url	URL	http://www.sas.com
:V	Verb	å, trikes, brukende, fyltes, brukte, krislende, brukt, gasjerer, slepp

Polish

Table A1.21 *Part-of-Speech Tags for Polish*

Part-of-Speech Tag	Description	Examples
:A	Adjective	własne, każda, głównych
:ABBREV	Abbreviation	p.n.e., n.e.
:ADV	Adverb	więcej, tylko
:CONJ	Conjunction	i, czyli
:INTER	Interjection	ej, fuj, amen
:N	Noun	teorie, miejscach, Wojciech
:NUM	Numeral	siedmiu, tysięcy
:PART	Particle	też
:PPOS	Preposition	za, z, na, do
:PRO	Pronoun	się, sami, go, tobie

Part-of-Speech Tag	Description	Examples
:V	Verb	wiedzieć, dotarł
:date	Date	:01/01/2012, 12/12/17, 12-23-2001, 23-12-01
:time	Time	23:30:01
:digit	Number	12, -5, 23,45
:sep	Separator or punctuation	. , -
:url	URL	http://www.sas.com
:PN	Unknown or foreign proper noun	Achitophel, Trzciński, LP-vinyl
:inc	Unknown or foreign word	sapiens, ela544

Portuguese

Table A1.22 Part-of-Speech Tags for Portuguese

Part-of-Speech Tag	Description	Examples
:A	Adjective	confiável, feliz
:ADV	Adverb	belamente, felizmente
:CONJ	Conjunction	e, que
:DET	Determiner	alguns, cada, os, dessas, dum
:digit	Number	21
:inc	Unknown word	xrxx
:INTJ	Interjection	caramba, eh
:N	Noun	beleza, felicidade, cf, ibid
:PN	Proper noun	Brasil, Portugal
:NUM	Numeral	bilionésimo, cinco
:PPOS	Preposition	com, de, em, anti, circum
:PRO	Pronoun	me, nós, quem
:sep	Separator or punctuation	,
:url	URL	http://www.sas.com

Part-of-Speech Tag	Description	Examples
:V	Verb	garanto, garantir, garantindo, garantido

Romanian

Table A1.23 Part-of-Speech Tags for Romanian

Part-of-Speech Tag	Description	Examples
:A	Adjective	înalt
:ADV	Adverb	taman
:CONJ	Conjunction	şi
:DET	Determiner	un
:INTJ	Interjection	hei
:N	Noun	carte
:NUM	Number	trei
:PN	Proper noun	Elena
:PPOS	Preposition	pro
:PRO	Pronoun	eu
:PUNC	Punctuation	!
:V	Verb	zisesem
:inc	Unknown word or non-word	asdfqwert
:digit	Numerical digit	999
:url	Internet address	http://sas.com
:date	Numerical date	2017-07-31
:time	Numerical time	23:30:00

Russian

Table A1.24 Part-of-Speech Tags for Russian

Part-of-Speech Tag	Description	Examples
:A	Adjective	духовитый, красивая, лучших, который, баскервиллей
:ADV	Adverb	дальше, сколько-нибудь, где, сколькие, почём
:conj	Conjunction	если, и
:digit	Number	123, 12.3, 12.3.2003, 12/3/2013
:inc	Unknown word	геминг, analytics
:INTJ	Interjection	ах
:N	Noun	велосипед, история, малолетство, др, км, мартини, маэстро
:PN	Proper noun	Шевроле, Айдахо, Миа, Роханский, Сашина, Свердловск, Мария, Давыдович
:NUM	Number	один, десятью
:PTCL	Particle	бы, же
:PPOS	Preposition	до, вроде
:PRO	Pronoun	я, её, всяко
:sep	Separator or punctuation	, . !
:url	URL	http://www.sas.com
:V	Verb	менять, нажимает, кладите, плавала, адаптировав, вальсируя

Slovak

Table A1.25 Part-of-Speech Tags for Slovak

Part-of-Speech Tag	Description	Examples
:A	Adjective	všeobecné , verejnej
:ADV	Adverb	pravidelne, vyslovene

Part-of-Speech Tag	Description	Examples
:CONJ	Conjunction	ak , iba
:INTJ	Interjection	oj, stop
:N	Noun	doručení, partnerov, ul, Dr
:NUM	Numeral	štyritisíc, prvom
:PTCL	Particle	by, tiež
:PPOS	Preposition	o, v, pre
:PRO	Pronoun	si, Vám, vaše, jeho, uňho, ktoré, akékoľvek
:V	Verb	prinášame, budú, nespráva, využívať, nezaostávať, prešli, nemali, pozrite
:digit	Number	1.4, -10, +421
:sep	Separator or punctuation	. , /
:PN	Proper noun	Oetker, KEPe
:inc	Unknown or foreign word	newslettri
:url	URL or email	http://www.sas.com, info@slovakrail.sk
:time	Time	23:30:00
:date	Date	23/12/2012, 23-12-2012

Slovene

Table A1.26 Part-of-Speech Tags for Slovene

Part-of-Speech Tag	Description	Examples
:A	Adjective	prvi, črna
:ADV	Adverb	hmalu, daleč
:CONJ	Conjunction	ali, in
:INTJ	Interjection	bravo, ah
:N	Noun	dni, dogodka, itd.
:NUM	Numeral	dva, šest

Part-of-Speech Tag	Description	Examples
:digit	Number	20.3, 123
:PTCL	Particle	pa, spet
:PPOS	Preposition	v, za
:PRO	Pronoun	te, mi, vsak, kdo
:V	Verb	sta, uporablja, suspendirali, pozabite
:sep	Separator or punctuation	. : , «
:Prop	Proper noun	Maribor, Roglič
:date	Date	23/12/2012, 23-12-2012
:time	Time	23:30:00
:url	URL	http://www.sas.com, info@sas.com

Spanish

Table A1.27 Part-of-Speech Tags for Spanish

Part-of-Speech Tag	Description	Examples
:A	Adjective	confiable, feliz, hermoso
:Adv	Adverb	ahora, felizmente
:CONJ	Conjunction	ni, pero, y
:DET	Determiner	mi, nuestro, al, del
:digit	Number	21
:inc	Unknown word	xrxx
:INTJ	Interjection	hola
:N	Noun	belleza, felicidad, km, pág, sra
:PN	Proper noun	Chile, España
:PPOS	Preposition	con, de, en, por
:PRO	Pronoun	alguien, ellos, me, el, las
:sep	Separator or punctuation	,

Part-of-Speech Tag	Description	Examples
:url	URL	http://www.sas.com
:V	Verb	ayudan, ayudar, ayudando, ayudado

Swedish

Table A1.28 Part-of-Speech Tags for Swedish

Part-of-Speech Tag	Description	Examples
:A	Adjective	fört
:ADV	Adverb	väl
:CONJ	Conjunction	samt
:DET	Determiner	Ens, somlig
:NUM	Number	två
:INTJ	Interjection	hej
:N	Noun	bok, morse, st.
:PN	Proper noun	Øsel, Tove, Östmark, Viklund, Toshiba
:PPOS	Preposition	till
:PRO	Pronoun	honom, du
:V	Verb	varit, varande, varats, sedd, ses, såg, sågs

Tagalog

Table A1.29 Part-of-Speech Tags for Tagalog

Part-of-Speech Tag	Description	Examples
:A	Adjective	abalang, alisto
:ADV	Adverb	biglang, bakit
:CONJ	Conjunction	at, yamang
:DET	Determiner	ni, nina

Part-of-Speech Tag	Description	Examples
:INTJ	Interjection	hoy
:N	Noun	pusa, yarda
:NUM	Number	dalawa, walumpu
:PN	Proper Noun	Asya, Espanya
:PPOS	Preposition	sa, dahil
:PRO	Pronoun	akin, amin, iyo
:PTCL	Particle	ay
:V	Verb	kainin, tayuan, uminom
:url	URL	www.sas.com
:date	Date	2015-12
:digit	Number	1, 20
:sep	Separator or punctuation	. , »
:inc	Unknown Word	possibilities, tropical
:time	Time	23:59:59

Thai

Table A1.30 Part-of-Speech Tags for Thai

Part-of-Speech Tag	Description	Examples
:ADJ	Adjective	กตัญญู, กตัญญูกตเวที
:ADV	Adverb	กระง่องกระแง่ง, กระดิบๆ
:AUXVERB	Auxiliary verbs	ควรจะ, ต้อง
:CLAS	Classifiers	กก., กม.
:CONJ	Conjunction	ก่อน, จน
:DET	Determiner	ทั้ง, ทุก
:END	Particle used at the end of a question, command, or entreaty	ล่ะ, เหรอ
:INTERJ	Interjection	ชะชะ, ดูกร

Part-of-Speech Tag	Description	Examples
:NEG	Negation	มิใช่, ไม่
:NOUN	Noun	กงพัด, กฎหมายบ้านเมือง
:NUMBER	Number	สอง, เก้า
:PREF	Prefix	ปรา, อน
:PREP	Preposition	กว่า, ก่อนหน้า
:PRON	Pronoun	คนอื่นๆ, คนใด
:PROPLOC	Proper noun, location	กมลา, กรีซ
:PROPMISC	Proper noun, others	กุชชี่, คลีนิกซ์
:PROPNAME	Proper noun, person names	กปิลกาญจน์, กตัญญุตานนท์
:PROPORG	Proper noun, organizations	กรุงเทพธุรกิจ, กระทรวงมหหาดไทย
:PUNC	Separator or punctuation	" (…)
:SUFF	Suffix	สิ, เอย
:VERB	Verb	กทรรป, กรมเกรียม
:DEFAULT	Unknown words	Josephson, microbridge

Turkish

Table A1.31 Part-of-Speech Tags for Turkish

Part-of-Speech Tag	Description	Examples
:A	Adjective	iyi, zor
:ADV	Adverb	yine, zaten
:CONJ	Conjunction	veya, hem
:date	Date	12/30/2000, 12/30/00, 2000-30-12
:digit	Number	12.302.000, 5
:inc	Unknown word	wug
:N	Noun	kitap, insan
:NUM	Numeral	dokuz, onbir, beri

Part-of-Speech Tag	Description	Examples
:PN	Proper noun	Ayşe, Türkçe
:PRO	Pronoun	bunlar, kendi, onlar, sen, çok
:sep	Separator or punctuation	! . ,
:time	Time	12:30:00
:url	URL	sas.com, www.sas.com, http://www.sas.com
:V	Verb	diye, bilir, bilmek, bilse, bilmiş, bildi, bilmeli, biliyor, bilmekte, bil

Vietnamese

Table A1.32 Part-of-Speech Tags for Vietnamese

Part-of-Speech Tag	Description	Examples
:A	Adjective	an toàn, bận rộn, lịch sự
:ABBREV	Abbreviation	APEC, ANĐT, ĐTNN
:Adv	Adverb	bỗng chốc, chưa chừng
:Aux	Particle	chính
:C	Conjugation	dù rằng, hoặc là
:F	Foreign word	cà-rem, Ampe, ăng ten
:Int	Interjection	hỡi, ái chà, ô hay
:N	Noun	áo quần, cừu, cương vị
:Num	Numeral	2007, bảy, mươi n
:PreDet	Determiner	một số
:Prep	Preposition	cho, vào
:PN	Proper noun	Việt Nam, Trung Quốc
:Pro	Pronoun	tôi, chúng mày, chúng nó
:PUNC	Punctuation or symbol	! : ( ) @ , ...
:RelPro	Relative pronoun	ai nấy

Part-of-Speech Tag	Description	Examples
:V	Verb	ngưỡng mộ, lưu nghiệm
:DEFAULT	Unknown word	₫
:date	Date	20/2/2012, 2017–04–10
:time	Time	23:59:59, 14:44
:url	URL, pathname, and email address	www.sas.com

Appendix 2

Predefined Concept Priorities

About Priority Values for Predefined Concepts

Priority values are used to determine which matches are returned when overlapping matches occur. The default priority setting is 10.

Arabic

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.1 Predefined Concept Priorities for Arabic

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization	20
nlpPercent	18
nlpPerson	20
nlpPlace*	25
nlpTime	18

Chinese

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.2 Predefined Concept Priorities for Chinese

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

Croatian

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.3 Predefined Concept Priorities for Croatian

Predefined Concept	Priority Value
nlpDate	10
nlpMeasure	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10
nlpPercent	10
nlpPerson	11
nlpPlace*	12
nlpTime	10

Czech

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.4 Predefined Concept Priorities for Czech

Predefined Concept	Priority Value
nlpDate*	10
nlpMoney*	10
nlpNounGroup	9
nlpOrganization*	10
nlpPercent*	10
nlpPerson*	10

Predefined Concept	Priority Value
nlpPlace*	10
nlpTime*	10

Danish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.5 *Predefined Concept Priorities for Danish*

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	15
nlpOrganization	10
nlpPercent	10
nlpPerson*	20
nlpPlace	10
nlpTime	10

Dutch

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.6 *Predefined Concept Priorities for Dutch*

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20

Predefined Concept	Priority Value
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

English

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.7 Predefined Concept Priorities for English

Predefined Concept	Priority Value
nlpDate	18
nlpMeasure	20
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	25
nlpPercent	18
nlpPerson	20
nlpPlace	20
nlpTime	18

Farsi

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.8 Predefined Concept Priorities for Farsi

Predefined Concept	Priority Value
nlpDate	18

Predefined Concept	Priority Value
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

Finnish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.9 Predefined Concept Priorities for Finnish

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	15
nlpOrganization*	25
nlpPercent	20
nlpPerson	20
nlpPlace*	25
nlpTime*	25

French

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.10 Predefined Concept Priorities for French

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

German

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.11 Predefined Concept Priorities for German

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	25
nlpNounGroup	15
nlpOrganization	25
nlpPercent	18
nlpPerson*	60
nlpPlace	40
nlpTime	18

Greek

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.12 Predefined Concept Priorities for Greek

Predefined Concept	Priority Value
nlpDate*	25
nlpMoney*	25
nlpNounGroup	15
nlpOrganization	20
nlpPercent*	25
nlpPerson	20
nlpPlace*	25
nlpTime*	25

Hebrew

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.13 Predefined Concept Priorities for Hebrew

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20

Predefined Concept	Priority Value
nlpTime	18

Hindi

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.14 Predefined Concept Priorities for Hindi

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10
nlpPercent	10
nlpPerson	10
nlpPlace*	40
nlpTime	10

Hungarian

For Hungarian, there are no specific priority values for predefined concepts. The default value of 10 is used for all of the predefined concepts listed below.

Table A2.15 Predefined Concept Priorities for Hungarian

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10

Predefined Concept	Priority Value
nlpPercent	10
nlpPerson	10
nlpPlace	10
nlpTime	10

Indonesian

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.16 Predefined Concept Priorities for Indonesian

Predefined Concept	Priority Value
nlpDate*	20
nlpMoney*	20
nlpNounGroup	10
nlpOrganization*	20
nlpPercent*	20
nlpPerson*	20
nlpPlace*	20
nlpTime*	20

Italian

For Italian, there are no specific priority values for predefined concepts. The default value of 10 is used for all of the predefined concepts listed below.

Table A2.17 Predefined Concept Priorities for Itlaian

Predefined Concept	Priority Value
nlpDate	10

Predefined Concept	Priority Value
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10
nlpPercent	10
nlpPerson	10
nlpPlace	10
nlpTime	10

Japanese

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.18 Predefined Concept Priorities for Japanese

Predefined Concept	Priority Value
nlpDate*	50
nlpMoney*	50
nlpNounGroup	20
nlpOrganization*	50
nlpPercent*	50
nlpPerson*	50
nlpPlace*	50
nlpTime*	50

Korean

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.19 Predefined Concept Priorities for Korean

Predefined Concept	Priority Value
nlpDate*	50
nlpMoney*	50
nlpNounGroup	35
nlpOrganization	40
nlpPercent*	50
nlpPerson	45
nlpPlace*	50
nlpTime*	50

Norwegian

For Norwegian, there are no specific priority values for predefined concepts. The default value of 10 is used for all of the predefined concepts listed below.

Table A2.20 Predefined Concept Priorities for Norwegian

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10
nlpPercent	10
nlpPerson	10
nlpPlace	10
nlpTime	10

Polish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.21 Predefined Concept Priorities for Polish

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	21
nlpPercent	18
nlpPerson	20
nlpPlace	20
nlpTime	18

Portuguese

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.22 Predefined Concept Priorities for Portuguese

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	25
nlpPercent	18
nlpPerson	20
nlpPlace*	25

Predefined Concept	Priority Value
nlpTime	18

Romanian

For Romanian, there are no specific priority values for predefined concepts. The default value of 10 is used for all of the predefined concepts listed below.

Table A2.23 Predefined Concept Priorities for Romanian

Predefined Concept	Priority Value
nlpNounGroup	10
nlpOrganization	10

Russian

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.24 Predefined Concept Priorities for Russian

Predefined Concept	Priority Value
nlpDate*	10
nlpMoney	9
nlpNounGroup*	10
nlpOrganization*	10
nlpPercent*	10
nlpPerson*	10
nlpPlace*	10
nlpTime*	10

Slovak

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.25 Predefined Concept Priorities for Slovak

Predefined Concept	Priority Value
nlpDate*	10
nlpMoney*	10
nlpNounGroup*	10
nlpOrganization*	10
nlpPercent*	10
nlpPerson	7
nlpPlace	8
nlpTime*	10

Slovene

For Slovene, there are no specific priority values for predefined concepts. The default value of 10 is used for all of the predefined concepts listed below.

Table A2.26 Predefined Concept Priorities for Slovene

Predefined Concept	Priority Value
nlpDate	10
nlpMeasure	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization	10
nlpPercent	10
nlpPerson	10

Predefined Concept	Priority Value
nlpPlace	10
nlpTime	10

Spanish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.27 Predefined Concept Priorities for Spanish

Predefined Concept	Priority Value
nlpDate	18
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	25
nlpPercent	18
nlpPerson	20
nlpPlace*	25
nlpTime	18

Swedish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.28 Predefined Concept Priorities for Swedish

Predefined Concept	Priority Value
nlpDate	18
nlpMeasure	18
nlpMoney	18
nlpNounGroup	15

Predefined Concept	Priority Value
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

Tagalog

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.29 Predefined Concept Priorities for Tagalog

Predefined Concept	Priority Value
nlpDate*	50
nlpMoney*	50
nlpNounGroup	35
nlpOrganization*	50
nlpPercent*	50
nlpPerson	40
nlpPlace	45
nlpTime*	50

Thai

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.30 Predefined Concept Priorities for Thai

Predefined Concept	Priority Value
nlpDate	18

Predefined Concept	Priority Value
nlpMoney	18
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime	18

Turkish

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.31 *Predefined Concept Priorities for Turkish*

Predefined Concept	Priority Value
nlpDate	10
nlpMoney	10
nlpNounGroup	10
nlpOrganization*	11
nlpPercent	10
nlpPerson	10
nlpPlace	10
nlpTime	10

Vietnamese

An asterisk (*) implies that a predefined concept has the highest priority value for this language.

Table A2.32 Predefined Concept Priorities for Vietnamese

Predefined Concept	Priority Value
nlpDate	18
nlpMoney*	20
nlpNounGroup	15
nlpOrganization*	20
nlpPercent	18
nlpPerson*	20
nlpPlace*	20
nlpTime*	20