Text Analytics Using SAS® Text Miner

Course Notes

Text Analytics Using SAS® Text Miner Course Notes was developed by Rich Perline based on revising and extending an earlier version written by Terry Woodfield. Additional contributions were made by Tom Bohannon, Peter Christie, George Fernandez, and Bob Lucas. Editing and production support was provided by the Curriculum Development and Support Department.

Text Analytics Using SAS® Text Miner Course Notes

Book code E2554, course code LWDMTX31/DMTX31, prepared date 18Nov2014. LWDMTX31_001

ISBN 978-1-62959-256-5

Table of Contents

Course Description

This course covers the functionality of SAS Text Miner software, which is a separately licensed component available for SAS Enterprise Miner. In this course, you learn to use SAS Text Miner to uncover underlying themes or concepts contained in large document collections, automatically group documents into topical clusters, classify documents into predefined categories, and integrate text data with structured data to enrich predictive modeling endeavors.

To learn more…

For information about other courses in the curriculum, contact the SAS Education Division at 1-800-333-7660, or send e-mail to training@sas.com. You can also find this information on the web at http://support.sas.com/training/ as well as in the Training Course Catalog.

For a list of other SAS books that relate to the topics covered in this course notes, USA customers can contact the SAS Publishing Department at 1-800-727-3228 or send e-mail to sasbook@sas.com. Customers outside the USA, please contact your local SAS office.

Also, see the SAS Bookstore on the web at http://support.sas.com/publishing/ for a complete list of books and a convenient order form.

Prerequisites

Before attending this course, you should have experience using SAS Enterprise Miner to do pattern discovery and predictive modeling, or you should have completed the Applied Analytics Using SAS® Enterprise Miner™ course.

A three-day version of this course contains the appropriate introductory material for using SAS Enterprise Miner. For the three-day course, you should also

- be acquainted with Microsoft Windows and Windows-based software
- have at least an introductory-level familiarity with basic statistics and regression modeling.

Previous SAS software experience, especially SAS Enterprise Miner, is helpful but not required. This course uses SAS Text Miner 13.1 and SAS Enterprise Miner 13.1.

Chapter 1 Introduction to SAS® Enterprise Miner™ and SAS® Text Miner

1.1 Data Mining and Text Mining

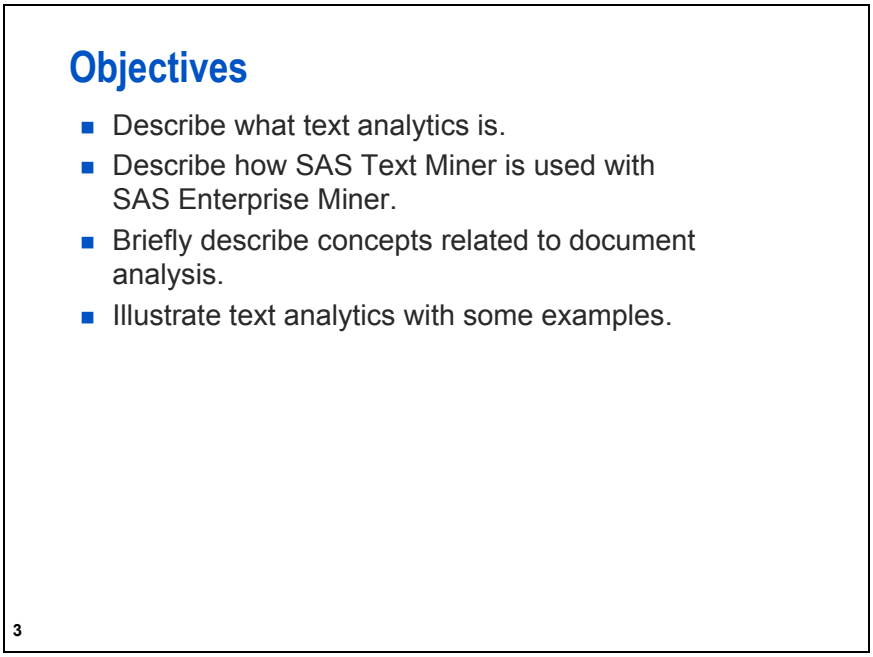
Preliminary Remarks

Text analytics is a vigorous field of research with many applications. The purpose of this course is to teach you how to solve analytic problems that include relevant textual data. This is done using SAS Enterprise Miner, a very general data mining product that incorporates text analytic tools among many other statistical and machine-learning tools.

Access to real business data is always problematic. Because text fields often contain confidential information, access to business data that includes text is even more difficult. Most data sets used in this course are publicly available. ***All data used in this course is either artificially created or modified in some way.*** Modifications include the following:

- deletion of sensitive entries
- deletion of potentially embarrassing or misleading entries
- editing or deletion of entries with named individuals or business organizations
- editing of text fields having obscure or confusing references
- resolution of ambiguities that might be incorrect
- modification or deletion of entries to promote educational goals

Because of these modifications, the data should not be used for any purpose other than education. All publicly available data sets are introduced with a reference to the source of the actual data. You should acquire data directly from the source if you want to use the data for business or scientific purposes.



Text Analytics

- You use the terms *text analytics*, *text data mining,* and *text mining* synonymously in this course.
- Text analytics uses ***algorithms*** for turning free-form text into data that can then be analyzed by applying ***statistical and machine learning methods***, as well as ***natural language processing techniques***.
- Text analytics encompasses many sub areas.

4

This course focuses on the use of SAS Text Miner, which can be directly integrated into SAS Enterprise Miner. SAS has a rich set of other text analytic products, but SAS Text Miner can be regarded as the most focused on discovery and prediction.

Visit **http://support.sas.com** for information about the latest text analytic offerings. Other courses present topics related to products such as SAS Enterprise Content Categorization.

This course discusses the two major components of data mining, ***pattern discovery or exploratory analysis*** and ***predictive modeling***, as they pertain to text analytics. It is helpful to describe some characteristics of text mining before you perform it.

Text Mining

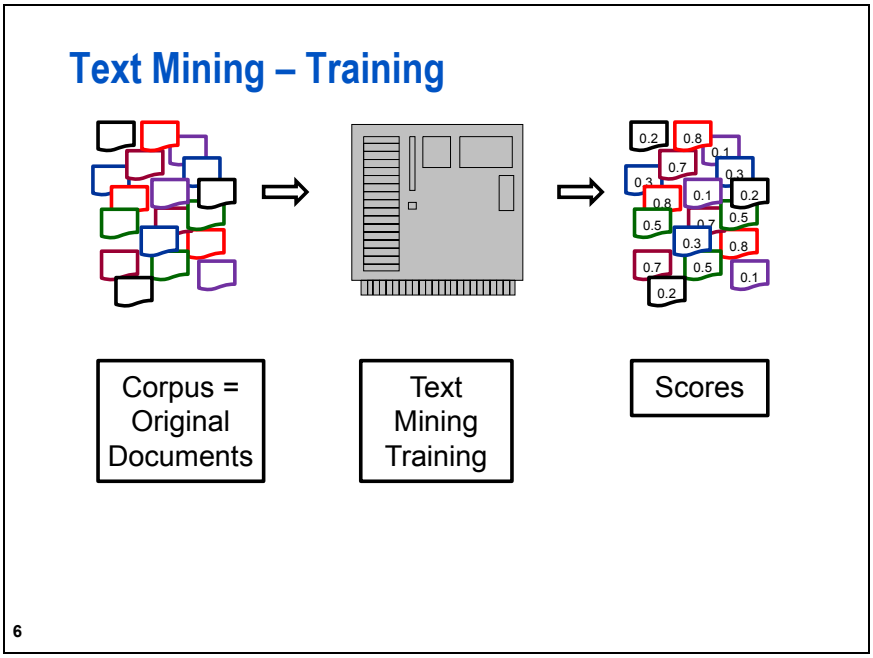
Text mining as presented here has the following characteristics:

- operates with respect to a ***corpus*** of documents
- creates a ***dictionary*** or ***vocabulary*** to identify relevant terms
- accommodates a variety of ***metrics*** to quantify the contents of a document within the corpus
- derives a ***structured vector*** of measurements for each document relative to the corpus
- uses ***analytical methods*** that are applied to the structured vector of measurements based on the goals of the analysis (for example, groups documents into segments)

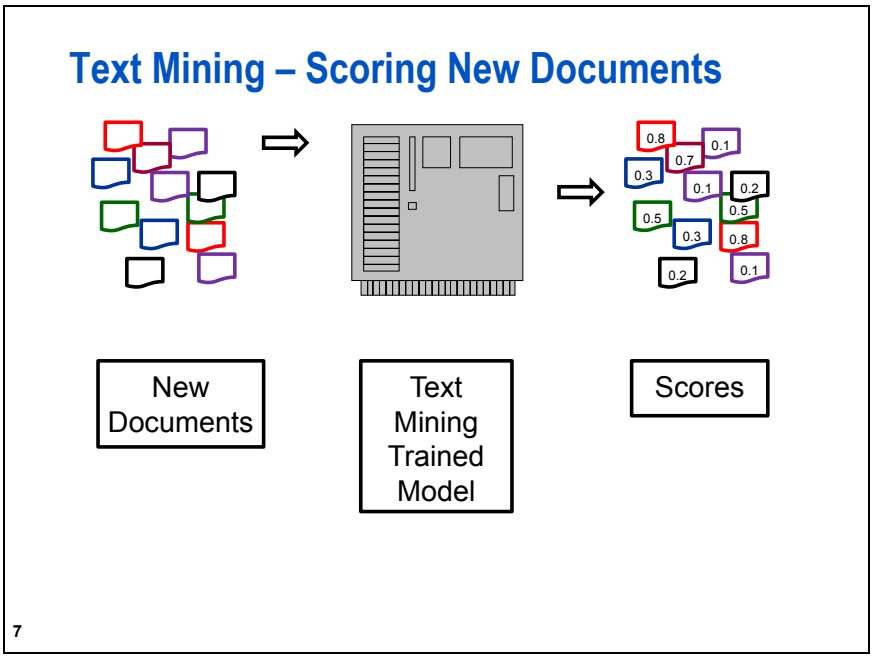
5

The concept of a dictionary can be thought of as a *vocabulary*. The document collection has a vocabulary that is the union of all the terms contained in each document. Consequently, text mining references do use ***dictionary*** or ***vocabulary*** to refer to the collection of terms that are used in the analysis. Terms not in the dictionary are ignored, except possibly for use in determining the relative frequencies of terms in each document. Zipf's Law, discussed in a later chapter, helps identify terms in a dictionary that should be included in an analysis. SAS Text Miner refers to the derived dictionary or vocabulary to be used for an analysis as the relevant terms.

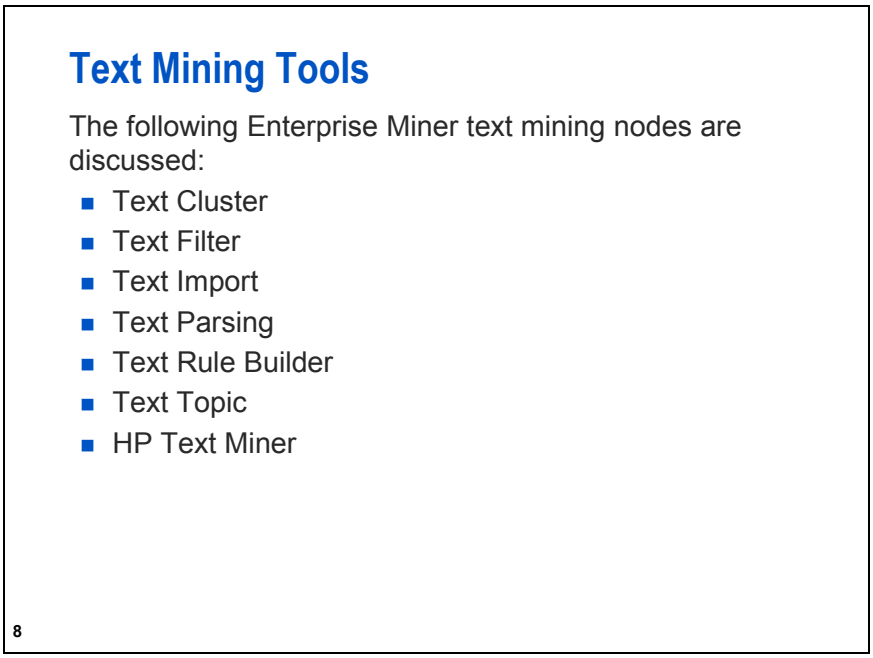
Text mining in this course works with a collection of documents. The collection can be dynamic, that is, documents can be added to the collection. You can use the collection to train a model, and you can apply the model to new documents coming into the collection. New documents are scored relative to how they compare to the original documents in the collection. If a new document contains a new term, then text mining is ignorant of this new term until that document is used in a new training step.



Text mining training can be performed using only nodes within the SAS Text Miner group of tools. However, SAS Text Miner nodes export data, and these data can be imported into pattern discovery and predictive modeling nodes of SAS Enterprise Miner. Thus, a trained model can be obtained by using a combination of SAS Text Miner nodes and SAS Enterprise Miner nodes. Although many commercial text mining products have strong text-analytics capabilities, most lack data mining capabilities beyond text analytics. The ability to score new documents using a decision tree or a neural network presents new opportunities to improve text mining outcomes (for example, making it possible to use variables derived from text analytics in predictive models).



As new documents appear, they can be scored using the model trained on the original corpus. Eventually, the model can be updated by being retrained on the corpus with the new documents added.



Other Enterprise Miner Nodes

You also use other Enterprise Miner nodes for various purposes, such as predictive modeling and scoring new cases.

- Data Partition node
- Decision Tree node
- Regression node
- Memory-Based Reasoning node
- Score node

9

Data mining analysts know how a predictive model scores new data. However, some analysts might be unaware that unsupervised learning models (that is, data without a known, available target) can also generate scores, and new data can be scored using the model. For example, the Text Cluster node divides a document collection into mutually exclusive clusters. A new document is scored by calculating the probability of membership in each cluster, and then it is assigned to the cluster associated with the highest probability.

Data mining is often described with respect to two general application areas: pattern discovery (unsupervised learning – no target variable) and predictive modeling (supervised learning – target variable). (Specific examples of these two application areas are presented in this course.)

Data Mining – Two Broad Areas

- Pattern Discovery/Exploratory Analysis (Unsupervised Learning)
  - There is no target variable, and some form of analysis is performed to do the following:
    - identify or define homogeneous groups, clusters, or segments
    - find links or associations between entities, as in market basket analysis
- Prediction (Supervised Learning)
  - A target variable is used, and some form of predictive or classification model is developed.
  - Input variables are associated with values of a target variable, and the model produces a predicted target value for a given set of inputs.

10

Text Mining Applications – Unsupervised

- Information retrieval
  - finding documents with relevant content of interest
  - used for researching medical, scientific, legal, and news documents such as books and journal articles
- Document categorization for organizing
  - clustering documents into naturally occurring groups
  - extracting themes or concepts
- Anomaly detection
  - identifying unusual documents that might be associated with cases requiring special handling such as unhappy customers, fraud activity, and so on

11

Anomaly detection can sometimes be a first step toward creating a target variable if none exists.

Text Mining Applications – Supervised

- Many typical predictive modeling or classification applications can be enhanced by incorporating textual data in addition to traditional input variables.
  - churning propensity models that include customer center notes, website forms, e-mails, and Twitter messages
  - hospital admission prediction models incorporating medical records notes as a new source of information
  - insurance fraud modeling using adjustor notes
  - sentiment categorization from customer comments
  - stylometry or forensic applications that identify the author of a particular writing sample

12

Text Mining Applications

The emphasis is predictive modeling applications, where free-form textual data can be used to derive new types of input variables.

- Predictive modeling requires data labeled with a known target (outcome) variable.
- Most analysts agree that predictive modeling is where the "big payoff" in data mining is.
- Predictive modeling takes most advantage of the integrated environment in SAS Enterprise Miner, which provides powerful predictive modeling tools (regression, decision trees, neural nets, and so on).

13

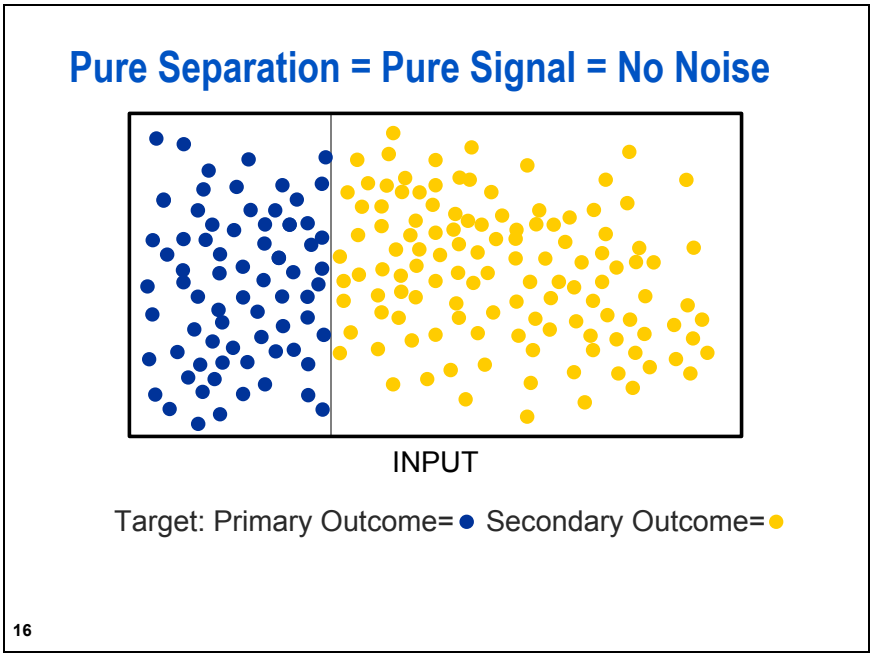
Signal versus Noise in Predictive Modeling

- Target = Signal + Noise
- Signal = Systematic Variation = Predictable
- Noise = Random Variation = Unpredictable

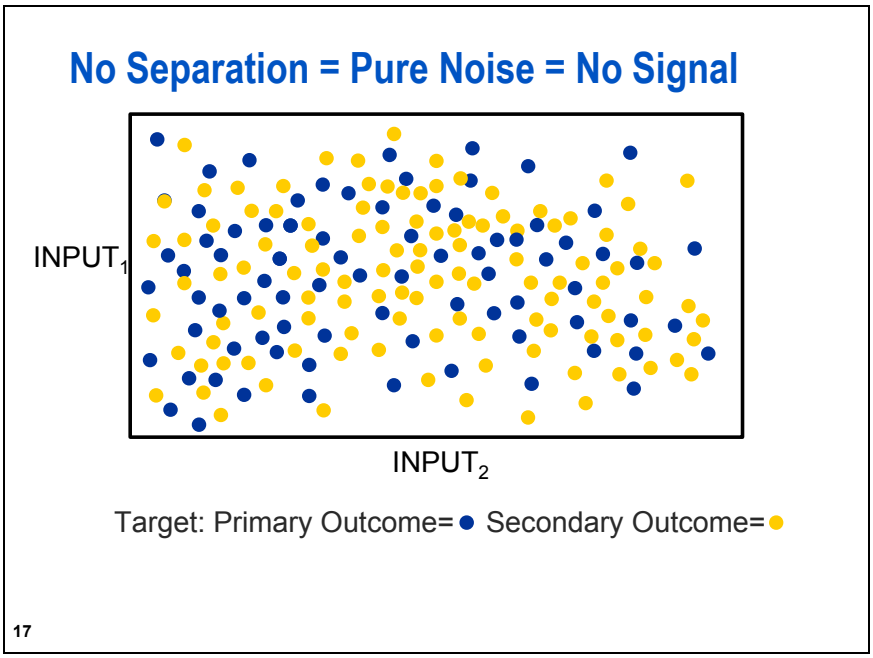
15

Users who are new to the world of analytics often have a naïve notion about noise. Science fiction movies include computers and robots that speak and understand human languages. Television police dramas have detectives who make perfect predictions about where crimes will occur. The reality is that noise permeates existence. You might have an expectation that, after you master text mining, you can perfectly predict customer behavior based on responses to an online survey. This expectation is unrealistic.

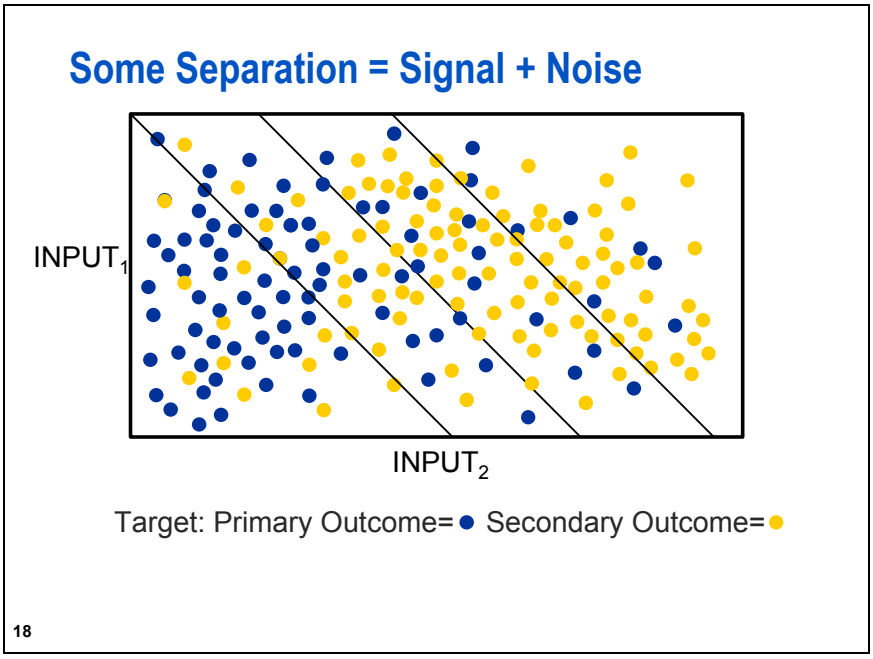
Psychologists know that human beings might react differently to the same stimulus if sufficient time elapses between exposures. On Monday, when you are hungry at lunchtime, you eat a sandwich. Yet, on Tuesday when you are hungry, you opt for a salad. This tendency for different outcomes to occur with similar inputs is attributed to noise, which is unpredictable. You can predict with almost certainty that you will eat lunch next Thursday, but you cannot predict what you will eat with the same certainty. (Of course, if you bet someone a million dollars that you will eat a spinach salad next Thursday, then you will almost certainly eat a spinach salad!) Analytic experts expect errors in prediction related to noise, so methods are developed to minimize errors in the presence of noise. The incremental value that text mining can provide to your predictive models should be assessed by comparing the quality of a model (accuracy, ROC index, and so on) without incorporating text mining to that achieved after text mining is added.



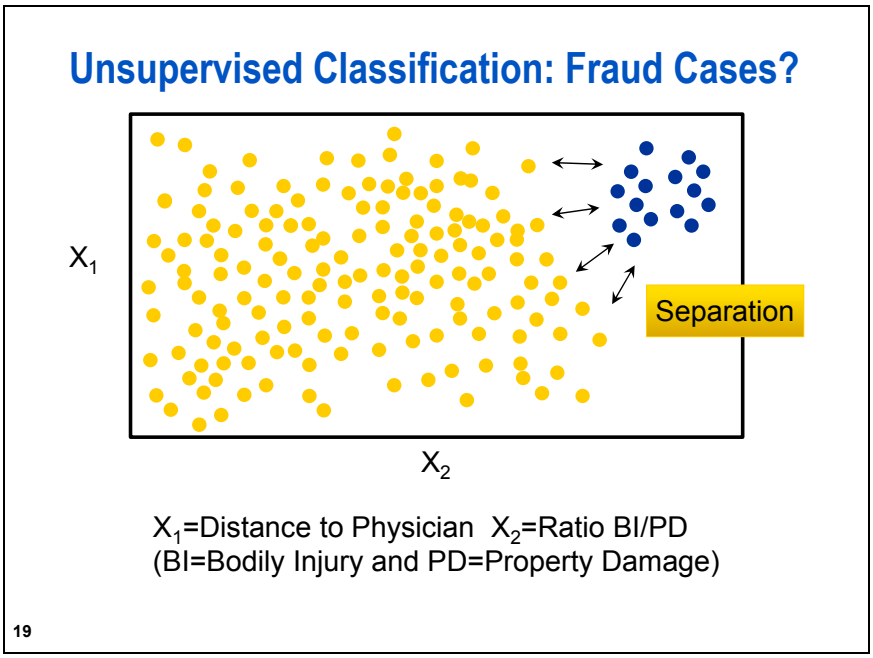
The above graphic illustrates the pure signal situation. In this case, the training data can be perfectly separated into primary or secondary outcomes using a linear decision boundary. You rarely expect to see this in practice. Unfortunately, some people who are new to text mining are disappointed when the methods do not perfectly categorize documents with this type of accuracy.



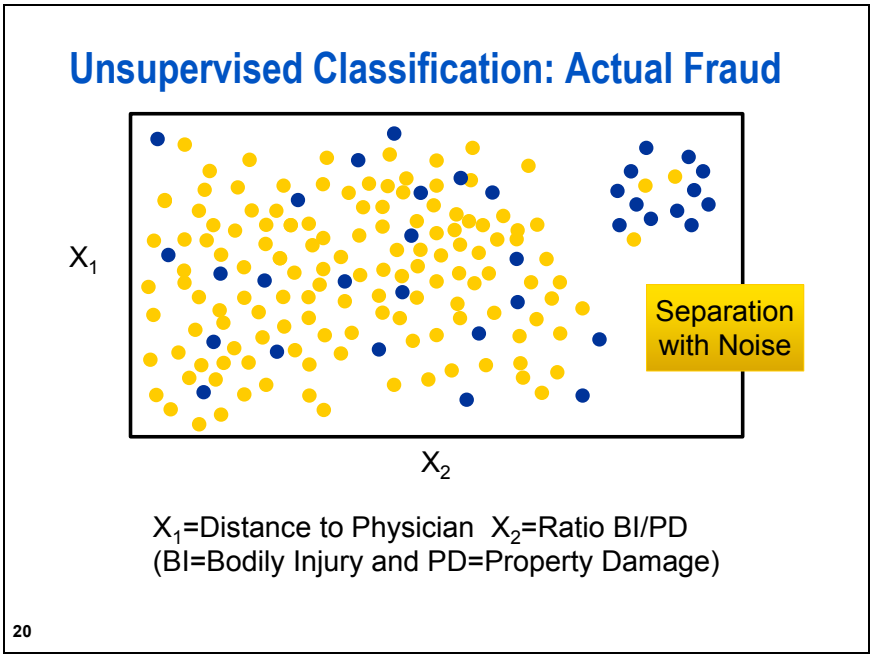
At the other extreme is the pure noise situation. In this case, the training data appears to have no patterns upon which to base a model that can separate the primary outcomes from the secondary outcomes. This situation is more common than you might like. Although pure signal is very rare, pure noise can actually occur in practice.



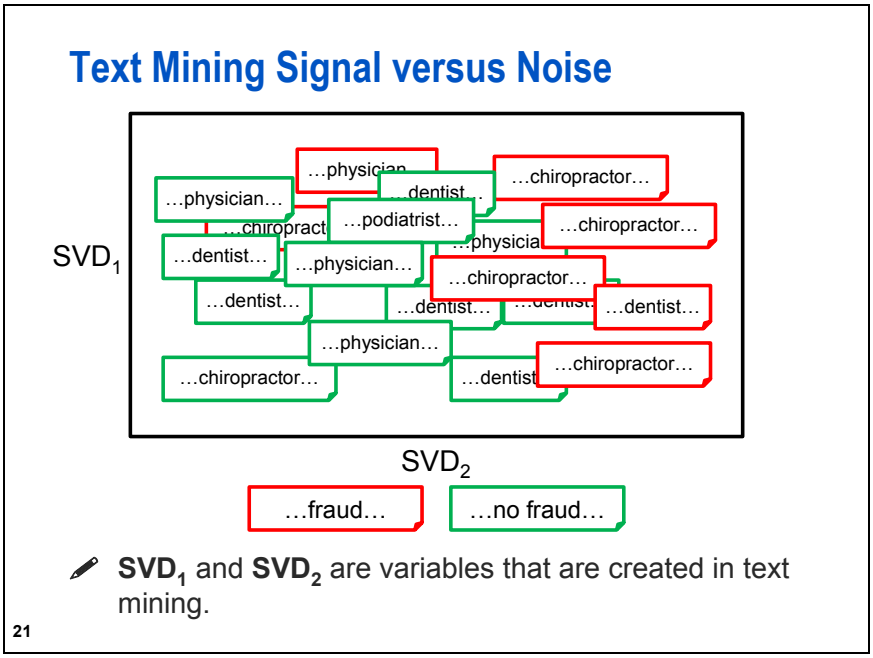
The most common situation in practice is a mixture of signal and noise. You can predict more accurately than randomly guessing. How well you predict depends on whether data is dominated by systematic variation or random variation.



When no target variable is available, you can still investigate whether a natural separation occurs in the data with respect to the analytic objective. For example, fraud cases are often unusual in higher dimensional space because the human beings that commit fraud have difficulty controlling outcomes so that they look normal in many dimensions. This example could represent insurance claims data for automobile accidents.



Even with good separation, noise is usually present. For the fraud example, most of the claims with a long distance from claimant to physician and a high ratio of bodily injury to property damage costs are fraudulent (dark circle), but a few are legitimate cases (light circle). Other fraud cases are not so separated, perhaps because the fraudulent physician had a practice near the claimant's home. In this example, BI is some quantitative measure of bodily injury and PD is a quantitative measure of property damage.



A string of fraud rings operating in Southern California in the 1990s had the common elements of a lawyer, a chiropractor, and a recruiter. The recruiter approached people who received unemployment benefits and told them that they could obtain worker's compensation benefits from their previous employers. The recruiter referred a candidate to an unscrupulous lawyer, who scheduled treatments with a chiropractor, a partner in the fraud ring. After a few weeks of treatments, the lawyer filed a claim for three to five times the chiropractor bills (a fairly common practice in insurance litigation). Claims adjusters often receive training in fraud prevention. When news of the fraud rings was disseminated, a claims adjuster might add a comment to the adjuster notes when unusual activity involving claimant representation by a lawyer and incoming chiropractor bills became known.

The above slide illustrates the following:

- The notes mentioned that chiropractors tend to be color-coded to indicate fraud.
- The notes mentioned that other medical professionals, like dentists, tend to be legitimate.

The two variables, **SVD$_1$** and **SVD$_2$**, are derived from running the Text Cluster node. (These are discussed later in this course.) Notice that, in this example, the fraudulent cases tend to be higher for the **SVD$_2$** variable.

Text Mining: Perfect Document Separation

Document ID	National News # Words	International News # Words	Document Subject
1	3	0	National
2	5	0	National
3	7	0	National
4	8	0	National
5	0	4	International
6	0	5	International
7	0	3	International
8	0	7	International

Perfect Separation: No Mixing of Subjects

22

Some document collections are well separated for analytic purposes. The hypothetical example above shows eight documents, with four that describe national news items exclusively, and the remaining four describing international news items exclusively. Suppose that you could identify a set of terms that are associated with national news and another set of terms associated with international news. These terms could then be used to classify the documents in the corpus.

Text Mining: Good but Imperfect Separation

Document ID	National News # Words	International News # Words	Document Subject
11	3	1	National
12	8	2	National
13	7	6	Mixed
14	8	1	National
15	1	4	International
16	2	5	International
17	3	3	Mixed
18	1	7	International

Good Separation: Little Mixing of Subjects

23

With the same topic and analytic objective, another document collection has documents that might mention a heterogeneous set of news articles. You still get good separation, but noise creeps in due to the fact that a document can include multiple subjects.

Text Mining: Poor Separation

Document ID	National News # Words	International News # Words	Document Subject
21	3	4	Mixed
22	8	2	National
23	7	6	Mixed
24	8	1	National
25	4	4	Mixed
26	6	5	Mixed
27	3	3	Mixed
28	1	7	International

Poor Separation: Substantial Mixing of Subjects

24

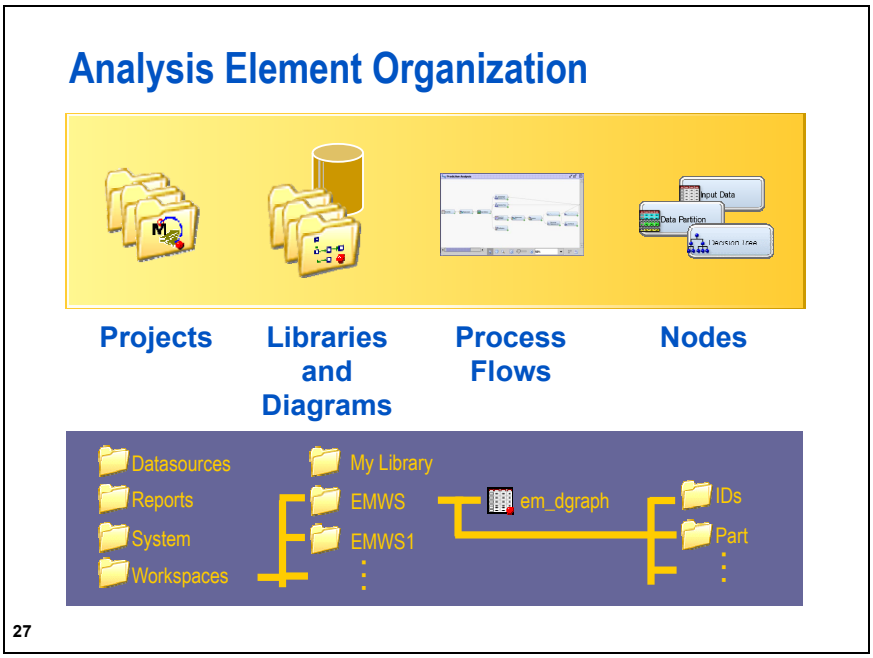
Finally, the above example shows that if you have a collection of documents that mention many topics and mixes topics, then trying to classify documents into clean categories is difficult.

1.2 Working with Data Sources

Objectives

- Describe SAS Enterprise Miner metadata and detail the types of roles and measurement levels that are supported.
- Explain how to create data sources that can be used by SAS Enterprise Miner projects.
- Provide examples of data sources that are relevant for text mining.

26

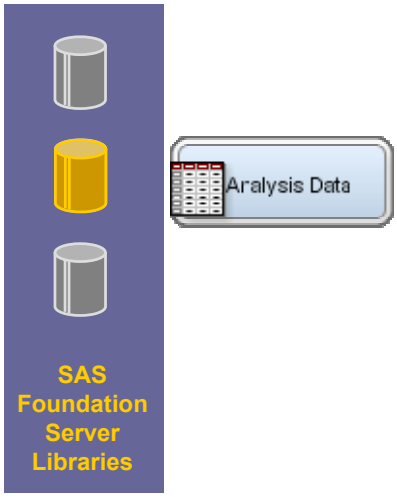


SAS Enterprise Miner organizes projects by placing components of the project in separate folders or directories. The Datasources folder contains metadata for each data source. The Workspaces folder holds all of the details about each diagram, including property settings of nodes used in each process flow.

SAS Enterprise Miner can import data from many sources, including common PC file formats such as Microsoft Excel and common commercial relational databases (for example, Sybase, Teradata, and Oracle), as well as from SAS data sets. The functionality of SAS Enterprise Miner comes from the assignment of roles and levels to variables in a data set. Initially assigning metadata roles makes the building of process flows much easier. Data properties do not need to be repeated or copied and pasted for each new task.

One of the first tasks in any project is to identify one or more relevant data sources. Although you can merge tables inside SAS Enterprise Miner, a best practice is to use the query optimization features of the native database to build the analysis table, and then import this table into SAS Enterprise Miner.

Defining a Data Source



- Select a table.
- Define variable roles.
- Define measurement levels.
- Define the table role.

28

Variable roles specify how a variable can be used.

Variable Roles

- Assessment
- Censor
- Classification
- Cost
- Cross ID
- Decision
- Frequency
- ID
- Input
- Label
- Prediction
- Referrer
- Rejected
- Residual
- Segment
- Sequence
- Target
- Text
- Text Location
- Time ID
- Web Address

29

These are the most common roles in text mining:

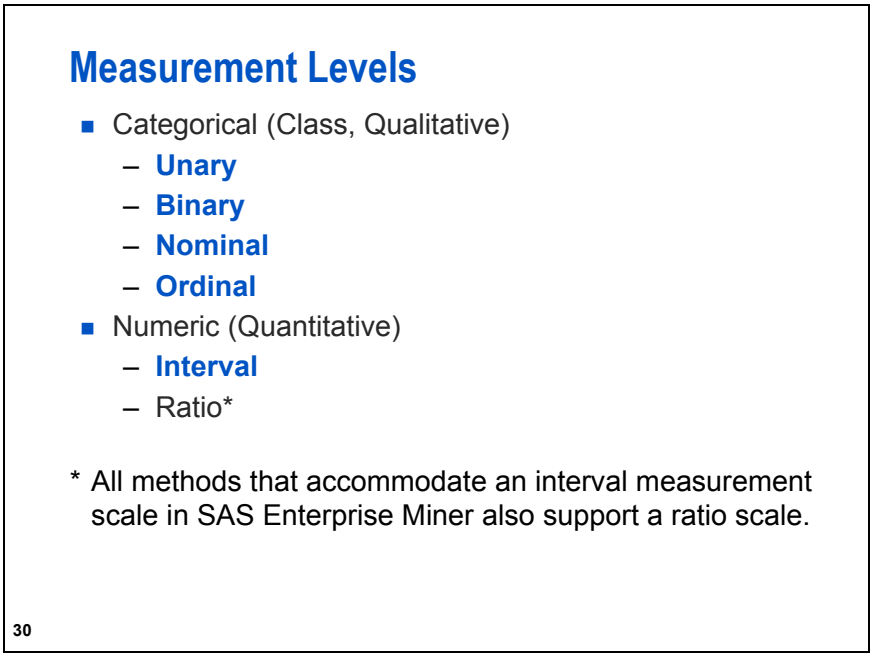
- Text
- Text Location
- ID
- Input
- Target
- Web Address
- Rejected

The document is stored as a variable with a role of *Text*. If the document is larger than 32,767 characters, and you want to analyze the entire document, then using *Text Location* is the way to do that due to the limitation of SAS variables to a maximum length of 32K bytes.

SAS allows 32,767=2**15-1 characters in a character variable. A document is stored as a SAS character variable in a SAS data set. The SAS Text Miner documentation rounds this down to 32,000, but 32,767 is the correct figure.

Additional variables in the data set usually have roles of ID, Input, Target, or Rejected. An *ID* variable identifies the document uniquely. An *input* variable can be used for segmentation or predictive modeling. Only input variables are used to derive segments or clusters. SAS Text Miner converts each document into a collection of inputs. For predictive modeling, the goal is to predict the value of a *target* variable. Only input variables are used to predict the target.

Any other variable in the data that has no purpose for the analysis has a role of *Rejected*.



Elementary statistics textbooks for social science majors usually describe four measurement levels:

- nominal
- ordinal
- interval
- ratio

Other statistics textbooks might speak only of categorical and numeric data.

A variable with a *nominal* measurement scale is purely categorical in nature. There is no numeric interpretation, and there is no natural ordering. Examples include eye color, political party affiliation, and country of origin. An *ordinal* variable is a categorical variable that has an inherent ordering. Thus, ordinal variables are also called *ordered categorical variables*. Examples include course letter grade, response on a Likert scale, or items on a top-10 ranking list.

Nominal data can be ranked by frequency of occurrence, price, personal preference, and so on. If the ranking is meaningful and exploited by the analysis, the nominal variable becomes an ordinal variable.

A binary scale implies a nominal scale with only two distinct values.

A variable with an *interval* measurement scale has a numeric interpretation so that the difference between two numeric values is meaningful. A variable with a *ratio* measurement scale is valid as an interval-scaled variable, but in addition, the ratio of two numeric values is meaningful. Temperature in degrees Celsius is on an interval scale, but not a ratio scale, because 20 degrees divided by 10 degrees being equal to 2 does not mean that 20 degrees are twice as hot as 10 degrees.

Most numeric data is on a ratio scale. Most analytic methods only require that the data be on an interval scale. All of the methods used by SAS Enterprise Miner that work for numeric data also work for interval or ratio-scaled values. Consequently, the ratio scale is not supported.

Different nodes expect specific table roles. The Score node scores raw, training, validation, test, and score data sets. The Association node acts on transaction data sets.

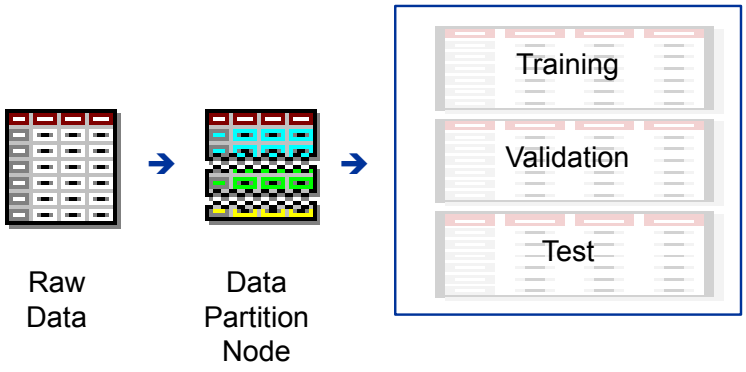
Table Roles

- Raw
- Training
- Validation
- Test
- Score
- Transaction

31

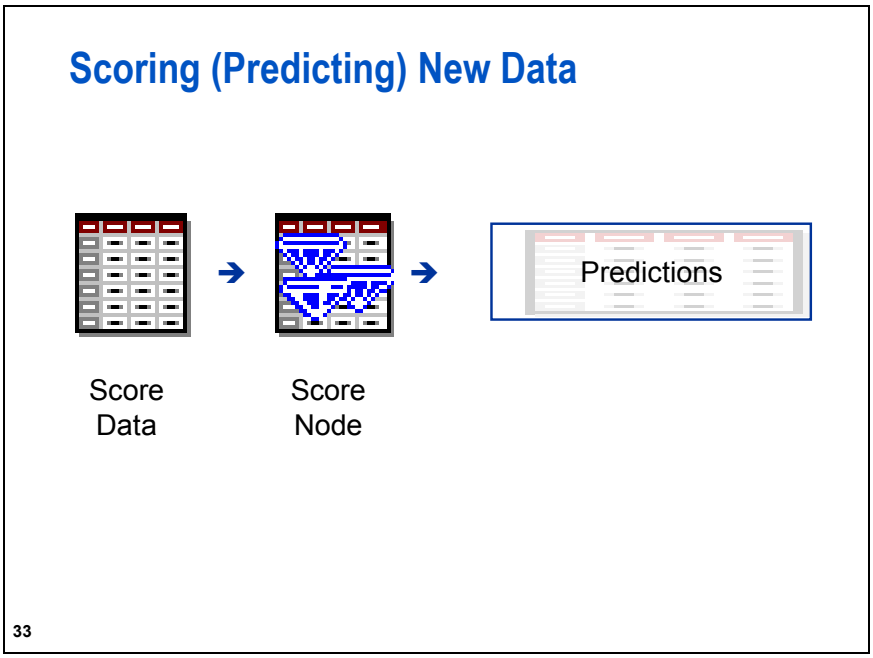
Using the Data Partition node, you can split raw data into training, validation, and test data sets. This is an important step in predictive modeling. You want to achieve good generalizability of the model by avoiding the problem of overfitting, that is, creating a model that looks good on the training data but does not generalize well to a holdout sample.

Analysis Data

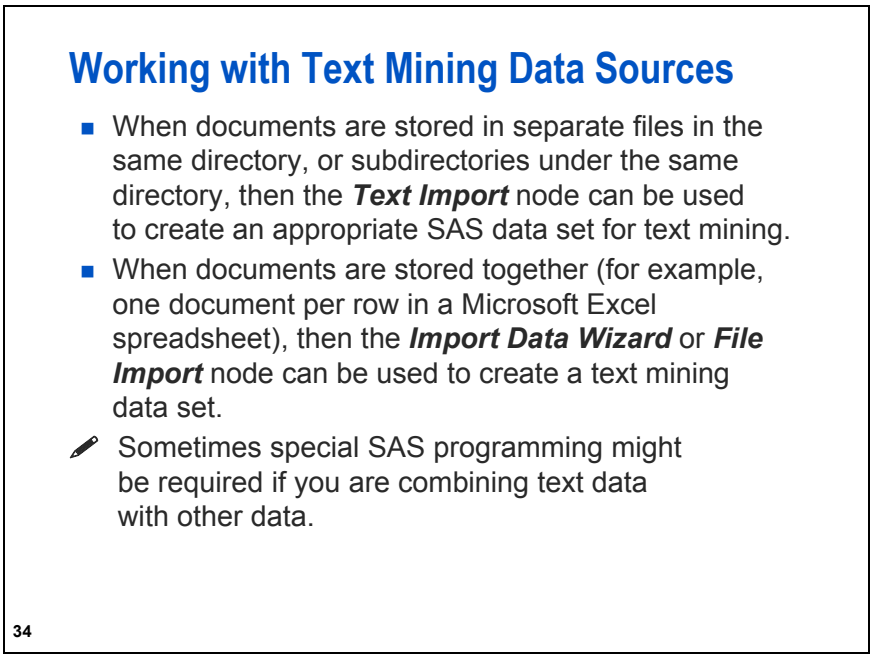


32

Score data can be scored by the Score node if all of the required data elements are present. The role of the score produced by the Score node is *Prediction* or *Segment*, depending on how the score is produced. Consequently, you need to be familiar with variable roles even if they are not assigned by you.



SAS Enterprise Miner and SAS Text Miner anticipate the need to create and modify data before an analysis. For text mining, the Text Import node can be deployed in a SAS Enterprise Miner process flow to process a document collection. (The Text Import node is discussed in the next chapter.)



The next slide describes how text data is treated by SAS Enterprise Miner.

Working with Text Mining Data Sources

Two supported types of text mining data:

- The data set contains at least one variable with the role **Text**, and documents can be stored completely as a SAS character variable (limited to 32K).
- The data set contains at least one variable with the role **Text Location**. (This will be used in the situation where a document size exceeds 32K.)
  - The location must be the full pathname of the document with respect to the Text Miner server.
  - An additional variable with the role **Web Address** can include the path to an unfiltered version of the document to be displayed in an interactive viewer such as the Interactive Filter Viewer.

35

Text parsing is always required as the first step in a text mining flow. This step accepts data sets with the role of Train, Validate, Test, or score data. At least one data source must be a data set with the role of Train or Raw.

The input data source must have at least one variable with a role of Text or Text Location. As stated above, the Text variable can contain an entire document or a truncated piece of an entire document. The Text variable is a character variable, and SAS can accommodate only character variables with lengths up to 32K (32,767 bytes). If a document exceeds 32K in length, then SAS must read the entire document from a location specified in the input data. If no location is specified, then the Text Miner nodes only process the truncated documents.

To process documents that exceed 32K, a variable with the Text Location role must be included in the input data. The text location must be the full pathname of the document folder with respect to the Text Miner server. For example, a document might be visible on your Windows computer at this location:

S:\MyProject\MyDocuments\Doc1.txt

The Text Miner server might recognize the location as follows:

//Sdisk/MyProject/MyDocuments/Doc1.txt

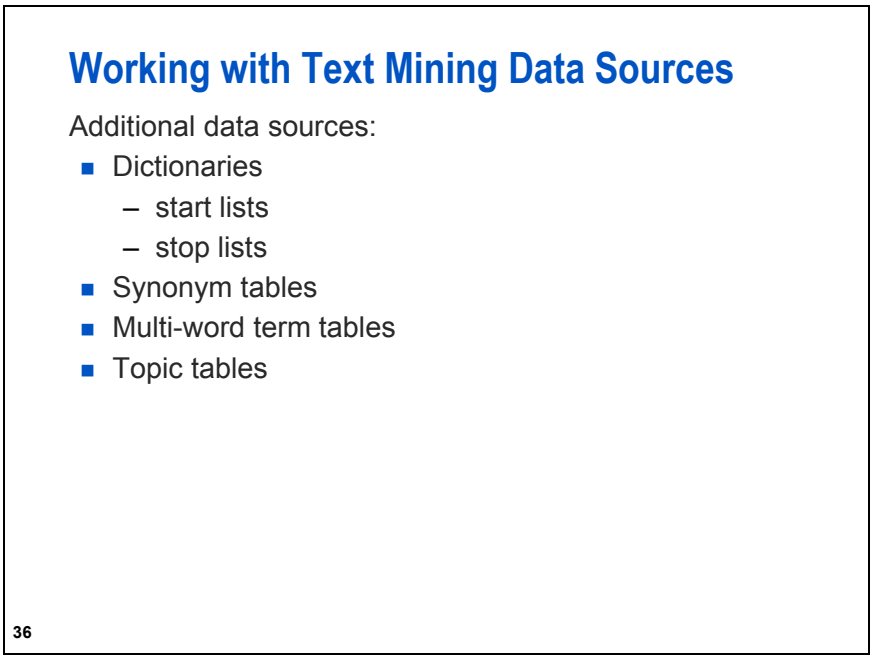
The second form of the document location must be used in the input data.

The Text Filter node can access documents through the Interactive Filter Viewer. By default, the Interactive Filter Viewer displays only the portion of the document stored in the Text variable. If you want to see the entire document in the Interactive Filter Viewer, then you can include a variable with the role of Text Location that provides the pathname of the file that contains the full document.

If the input data source contains two or more variables with a role of Text, and the Use status is *Yes* for these variables, then the Text Parsing node chooses the variable with the largest length. If the lengths are the same, then the variable that appears first in column order is selected. If your data has two or more text variables, you should set the Use status to *No* for all text variables except the one to be included in the analysis.

If you want to include two or more text variables in your text mining project, then you must connect Text Parsing nodes in parallel and change the Use status of the variables as needed.

In many cases, you need to preprocess textual data before you can import it into a data source. The Text Import node is designed for this purpose. The Text Import node can be used in file preprocessing to extract text from various document formats or to retrieve text from websites by crawling the web. The node creates a SAS data set that you can use to create a data source to use as input for the Text Parsing node. Depending on which structure (of the two described above) that you use, you must adjust the roles of the variables accordingly in the Data Source Wizard.



The software distribution of SAS Enterprise Miner includes the following sample data sets in the Sashelp library:

Data Set	Description	Used In
***language*_multi**	Multi-term lists for various languages	Text Parsing node
***language*stop**	Stop lists for various languages	Text Parsing node
Engstop	Stop list for the English language	Text Parsing node
Engsynms	Synonym list for the English language	Text Parsing node

The keyword *language* is chosen to correspond to one of these supported languages: Arabic, Chinese (simplified and traditional), Czech, Danish, Dutch, English, Finnish, French, German, Greek, Hebrew, Hungarian, Indonesian, Italian, Japanese, Korean, Norwegian, Polish, Portuguese, Romanian, Russian, Slovak, Spanish, Swedish, Thai, Turkish, and Vietnamese. (However, only English, French, German, Italian, Portuguese, and Spanish have built-in stop lists and multi-term lists.)

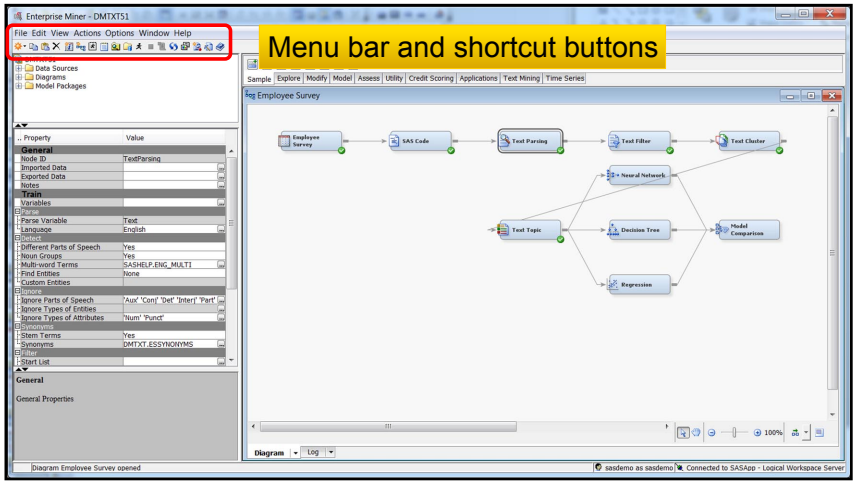
1.3 Using SAS Enterprise Miner and SAS Text Miner

Objectives

- Describe SEMMA data mining methodology.
- Tour the SAS Enterprise Miner user interface.
- Describe SAS Text Miner.
- Explain some of the nodes in SAS Text Miner.
- Use the SAS Text Miner nodes to explore a document collection.

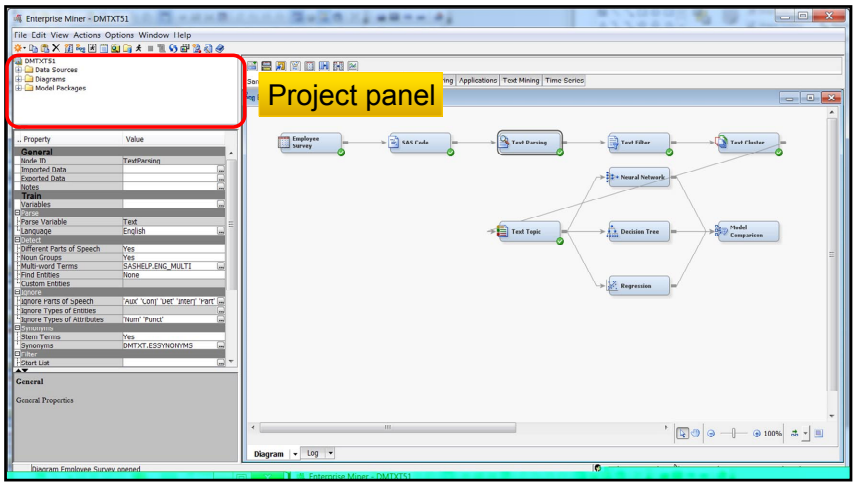
38

SAS Enterprise Miner: Interface Tour



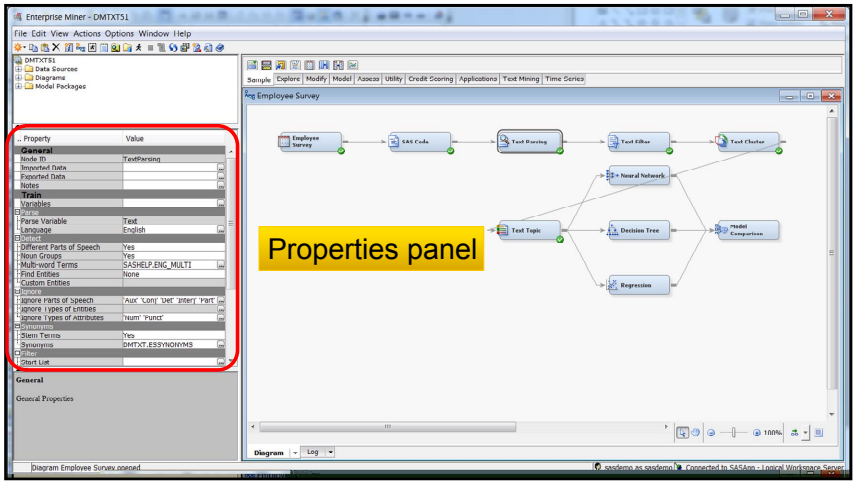
39

SAS Enterprise Miner: Interface Tour



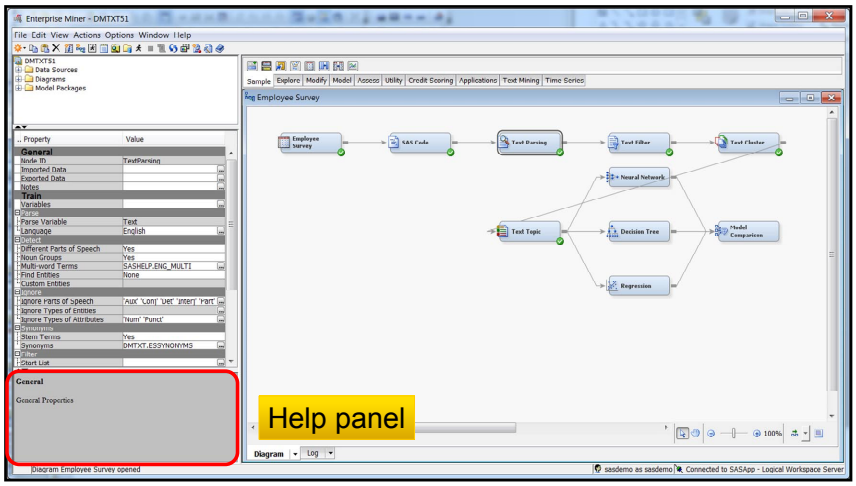
40

SAS Enterprise Miner: Interface Tour



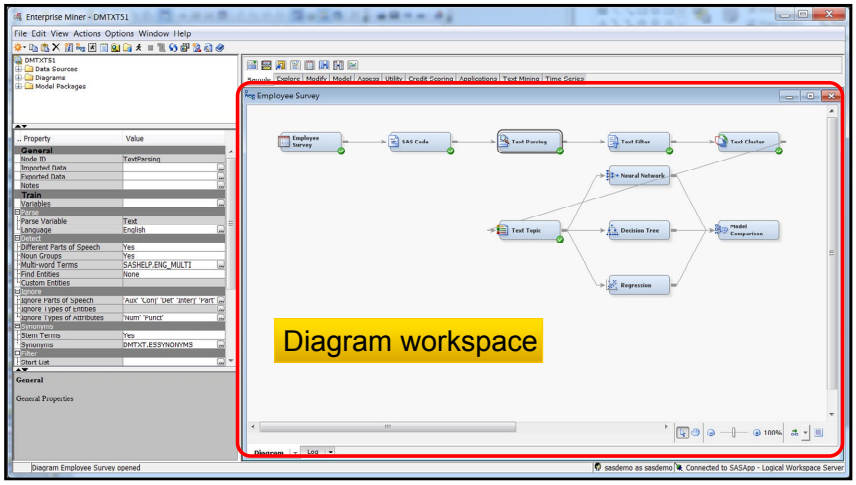
41

SAS Enterprise Miner: Interface Tour



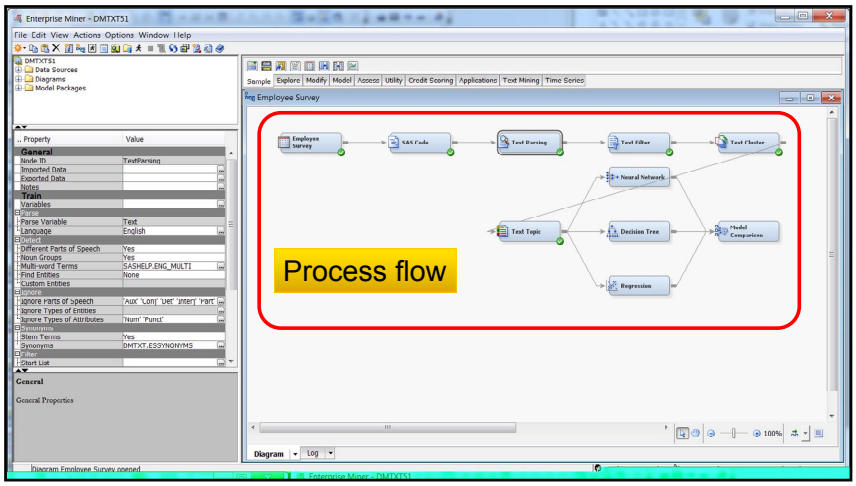
42

SAS Enterprise Miner: Interface Tour



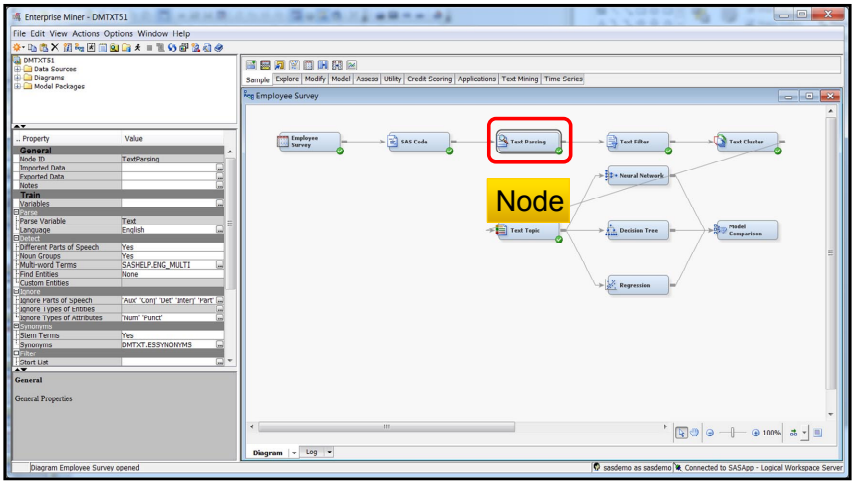
43

SAS Enterprise Miner: Interface Tour

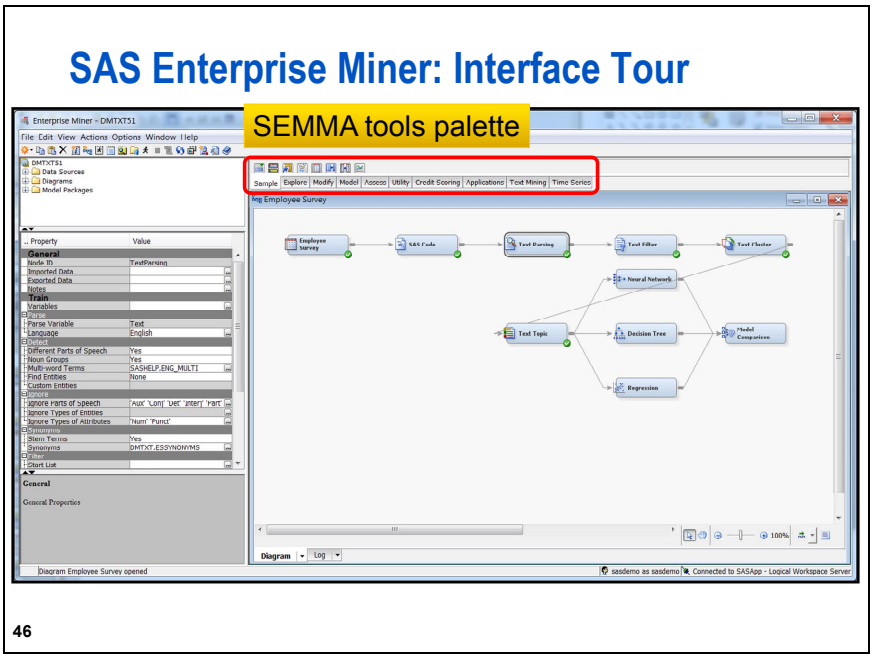


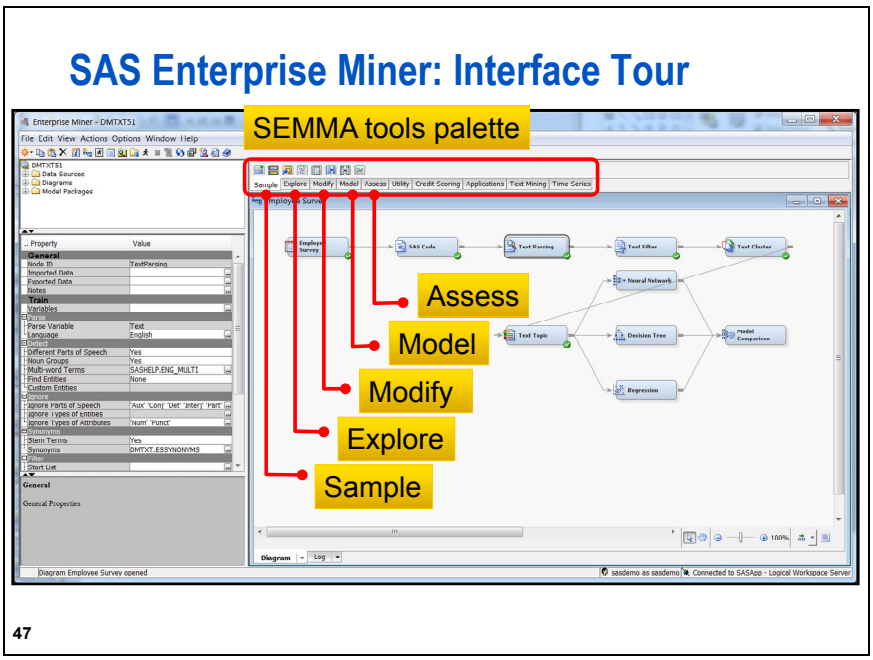
44

SAS Enterprise Miner: Interface Tour



45





The available SAS Enterprise Miner tools are contained in the *tools palette*. The most commonly used tools are arranged according to a process for data mining referred to as *SEMMA*. This is an acronym for the following:

Sample You sample the data by creating one or more data tables. The samples should be large enough so that you have confidence in the reliability of the results.

Explore You explore data to better understand relationships, anomalies, and problems.

Modify You modify the data by cleaning, selecting, and transforming the variables considered for modeling.

Model You model the data using the available analytical tools.

Assess You assess and compare alternative models to find the best results that you can obtain.

Additional tools (nodes) are available on the Utility tab. Additional nodes can be licensed, such as the Credit Scoring nodes and the Text Mining nodes.

SAS Enterprise Miner: Interface Tour

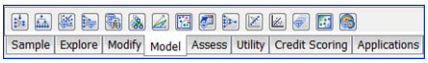
SEMMA tools palette

Sample

Explore

Modify

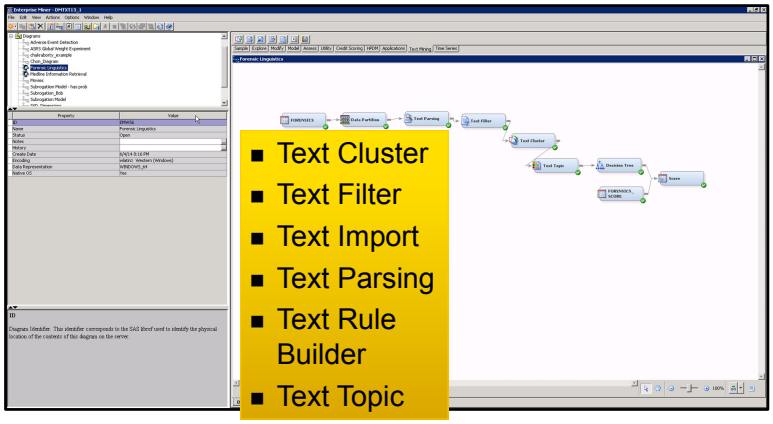
Model



Assess

48

Using SAS Text Miner



- Text Cluster
- Text Filter
- Text Import
- Text Parsing
- Text Rule Builder
- Text Topic

In this class, you also use the High Performance (HP) Text Miner node.

49

Using the Text Parsing Node

Text Parsing Properties

.. Property	Value
General	
Node ID	TextParsing
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Parse	
Parse Variable	
Language	English
Detect	
Different Parts of Speech	Yes
Noun Groups	Yes
Multi-word Terms	SASHELP.ENG_MULTI
Find Entities	None
Custom Entities	
Ignore	
Ignore Parts of Speech	'Aux' 'Conj' 'Det' 'Interj' 'Part'
Ignore Types of Entities	
Ignore Types of Attributes	'Num' 'Punct'
Synonyms	
Stem Terms	Yes
Synonyms	SASHELP.ENGSYNMS
Filter	
Start List	
Stop List	SASHELP.ENGSTOP
Report	
Number of Terms to Display	20000

50

The Text Parsing Node

The Text Parsing node

- builds the corpus dictionary
- associates terms with parts of speech and controls which parts of speech to recognize
- performs stemming to equate terms that are different verb tenses of the same verb, or to equate terms that are either singular or plural versions of the same noun
- identifies up to 16 entities such as address, company name, currency, and person's name
- imports custom entities created by a product such as SAS Concept Creation or SAS Content Categorization Studio
- controls recognition of numbers or punctuation as separate terms.

51

Custom entities are not discussed in this course. You can refer to SAS Text Miner 13.1 Reference Help for information about how to bring in results to the Text Parsing node from SAS Concept Creation for SAS Text Miner in SAS Content Categorization Studio.

The Text Parsing Node

Verb stemming example:

- type
- typed
- typing
- types

Noun stemming examples:

- house, houses
- matrix, matrices
- criteria, criterion

52

The Text Parsing Node

The Text Parsing node special tables:

- Synonyms
- Multi-word term dictionary
- Start/stop list - table of terms to include or exclude from the analysis

53

The Text Parsing Node

Dictionaries when a **stop list** is specified:

- **Corpus dictionary**: the union of all terms in the corpus (derived, not specified)
- **Stop list**: a dictionary of terms to be ignored in the analysis (specified by the user)
- **Start list**: terms in the corpus dictionary that are not in the stop list (derived)

The stop list is typically used to remove **low information** terms that add only **noise** to the analysis. **Noisy data** has no descriptive or predictive value.

54

The Text Parsing Node

Dictionaries when a **start list** is specified:

- **Corpus dictionary**: the union of all terms in the corpus (derived, not specified)
- **Start list**: a dictionary of terms to be used in the analysis (specified by the user)
- **Stop list**: terms in the corpus dictionary that are not in the start list (derived)

The start list can be a technical or business dictionary that is developed by the analyst or obtained from other sources.

55

Using the Text Filter Node

Text Filter Properties

.. Property	Value
General	
Node ID	TextFilter
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Spelling	
Check Spelling	No
Dictionary	
Weightings	
Frequency Weighting	Default
Term Weight	Default
Term Filters	
Minimum Number of Documents	1
Maximum Number of Terms	.
Import Synonyms	
Document Filters	
Search Expression	
Subset Documents	
Results	
Filter Viewer	
Spell-Checking Results	
Exported Synonyms	
Report	
Terms to View	All
Number of Terms to Display	20000

56

The Text Filter Node

Text Filter Properties

- Frequency weights and term weights are included.
- The optional Check Spelling property uses a spelling dictionary and word-similarity algorithms to find and correct misspellings.
- The Minimum Number of Documents property performs frequency filtering for rare terms. This property can be used rather than searching for rare terms and adding them to the stop list.
- The Filter Viewer enables you to interactively control terms to drop or keep, interactively create synonyms, and perform queries and view concept links.

57

The Text Filter Node

Analysis Features

- Frequency weights
  - Log (default)
  - Binary
  - None (count or frequency)
- Term weights
  - Entropy (default)
  - Inverse Document Frequency
  - Mutual Information
  - None

58

The Text Filter Node

Query filters have the following characteristics:

- can be used in the Properties panel and in the Interactive Filter Viewer
- return documents satisfying the query
- can be used to subset the document collection for the continuing downstream analysis of the collection

59

The Text Filter Node

Query Operators

- +*term* returns all documents having at least one occurrence of *term.*
- -*term* returns all documents having zero occurrences of *term.*
- "*text string*" returns all documents having at least one occurrence of the quoted text string.
- *string1*string2* returns all documents that have a term that begins with *string1*, ends with *string2*, and has text in between.
- >#*term* returns all documents that have *term* or any of the synonyms that are associated with *term.*

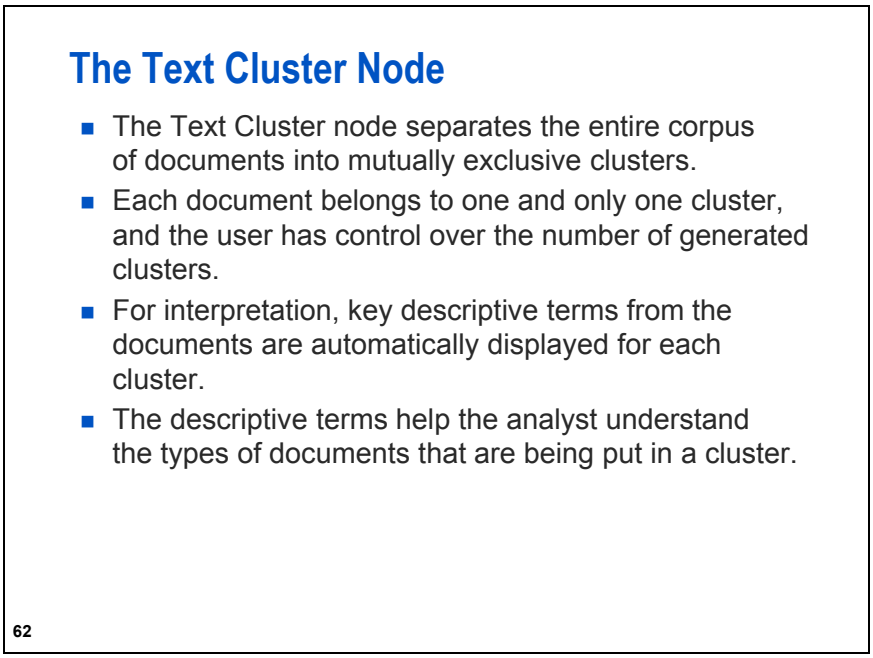
60

Using the Text Cluster Node

Text Cluster Properties

.. Property	Value
General	
Node ID	TextCluster
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Transform	
SVD Resolution	Low
Max SVD Dimensions	100
Cluster	
Exact or Maximum Number	Maximum
Number of Clusters	40
Cluster Algorithm	EXPECTATION-MAXIMIZATION
Descriptive Terms	8

61



The Text Cluster node divides a document collection into mutually exclusive clusters. By default, 15 terms are displayed that are most strongly associated with each of the clusters. These descriptive terms help the analyst understand the types of documents that are in a given cluster. In the first hands-on demonstration in the next section, you see that one cluster of documents has descriptive terms such as *cold*, *rain*, *snow*, and *winter*. This highlights the fact that this cluster consists mostly of documents about the weather. (It is possible that a descriptive term displayed for one cluster can also be important for describing other clusters.)

The interpretable value of the descriptive terms becomes clear as you work through some hands-on examples with the Text Cluster node.

From **Help** ⇨ **Contents**:

“The Text Cluster node uses a descriptive terms algorithm to describe the contents of both EM clusters and hierarchical clusters. If you specify to display m descriptive terms for each cluster, then the top 2*m most frequently occurring terms in each cluster are used to compute the descriptive terms.”

“For each of the 2*m terms, a binomial probability for each cluster is computed. The probability of assigning a term to cluster j is prob=F(k|N, p). Here, F is the binomial cumulative distribution function, k is the number of times that the term appears in cluster j, N is the number of documents in cluster j, p is equal to (sum-k)/(total-N), sum is the total number of times that the term appears in all the clusters, and total is the total number of documents. The m descriptive terms are those that have the highest binomial probabilities.”

“Descriptive terms must have a keep status of Y and must occur at least twice (by default) in a cluster.”

The Text Cluster Node

- For example, suppose the corpus of documents is a collection of newspaper articles – some of them about sports and others about politics. Then you might expect to see the following:
  - one cluster of documents with key descriptive terms such as *baseball*, *soccer*, *score*, and so on
  - another cluster of documents with key descriptive terms such as *election*, *campaign*, *votes*, and so on
- The Text Cluster node is run after
  - the Text Parsing node performs its natural language processing and "tokenized" the terms
  - the Text Filter node selects the terms to work with and applies certain weights.

63

The Text Cluster Node

- Clustering is performed by using a linear algebra approach to the term-document frequency matrix.
- As an example of the raw input data that ultimately is processed to produce clusters, the table below shows that "cat" occurred three times in Document 1 and "dog" occurred two times in Document 2.

	Doc 1	Doc 2	Doc 3	...	Doc N
apple	1	0	0	...	2
cat	3	1	1	...	4
dog	2	2	1	...	3
farm	1	0	0	...	1
...	...	...	...	...	...
White House	0	3	4	...	0
Senate	0	2	4	...	0

- Basically, documents that have similar term usage tend to be put in the same cluster. In this case, Document 2 and Document 3 look somewhat alike.

64

The use of *singular value decomposition* (SVD) is called the *linear algebra approach* to text mining, information retrieval, web analytics, and so on. As mentioned in a previous section, this algebraic operation is the foundation of an approach that has many names:

- Latent Semantic Indexing (LSI)
- Latent Semantic Analysis (LSA)
- Vector Space Model (VSM)

The Text Cluster Node

- The user also has control of the cluster derivation.
  - Choose the exact ***or*** maximum number.
  - Choose the maximum number of clusters.
  - Choose the cluster algorithm that is used.
    - Expectation-Maximization (EM)
    - Hierarchical
- Clustering documents is a powerful analytic approach, but can you see a possible shortcoming of this idea?
  - To use the previous example, what happens to a newspaper article that deals with both sports ***and*** politics?
  - It can be placed only in one cluster or another, but not both. That is why there is also a ***Text Topics*** node.

65

Using the Text Topic Node

Text Topic Properties

.. Property	Value
General	
Node ID	TextTopic
Imported Data	
Exported Data	
Notes	
Train	
Variables	
User Topics	
Term Topics	
Number of Single-term Topics	0
Learned Topics	
Number of Multi-term Topics	25
Correlated Topics	No
Results	
Topic Viewer	

66

The Text Topics Node

- A *topic* is a subject or theme or idea that occurs in a document.
- For example, suppose the document is a newspaper article containing topics related to the following:
  - sports
  - politics
  - law
- Clearly, a document can contain more than one topic (***whereas a document can belong only to one cluster***).
- Topics are generated
  - automatically by the Text Topic node using basically the same underlying mathematical algorithm that Text Cluster uses (modified a bit)
  - also by user definitions.

67

The Text Topics Node

- The basic idea behind automatic topic generation is to find terms that occur frequently together within documents. Together these germs “define” the topic.
- This approach can be looked at by thinking of terms as potential “friends” in a social network.
  - The terms *car* and *auto* might co-occur (be “friends”) within many of the same documents.
  - ***However***, even if they are not direct “friends” in the same documents, they can still be “friends of friends.”

68

The Text Topics Node

- As an example of this indirect connection, suppose *car* and *auto* ***never*** co-occur in the same document, but each co-occurs frequently with *tire*. Therefore, *car* and *auto* are friends of the same friend and might be recognized by the Text Topic algorithm as key descriptive terms for the same topic.
- By default, 25 topics are automatically generated.
- These topics are identified (or interpreted) by the analyst most frequently by examining a list of five key descriptive terms that are automatically displayed.

69

The Text Topic Node

Text Topic Properties

- A custom topic table can be supplied by the user. The table can be imported, or the table can be manually created with a table editor.
- The user can request up to 1,000 single-term topics to be derived. By default, no single-term topics are derived.
- A user can specify up to 1,000 multi-term topics to be derived. By default, 25 multi-term topics are derived.

70

The Text Topic Node

Topics

- Single-term topics are not the same as filtering on a single term. For example, a topic can be derived based on the single-term *price*, but documents might be labeled as not having the topic even if the term *price* is present in the document.
- The node might return fewer topics than requested. After the designated number of topics are derived, the node can decide that topics 24 and 25 are not sufficiently distinct to warrant including both topics, for example. If so, topic 25 (based on order of importance) is dropped.

71

The Text Topic Node

Custom Topics

- A custom (user-defined) topic consists of a label and one or more terms. Each term has a role and a weight.
- The weight associated with a term-role pair indicates the analyst's judgment about the relative importance of the term-role pair to the topic.
- In practice, most users define weights in the range of 0 to 1, where 1 is the highest importance and a weight of 0 is the lowest.

72

Using the Text Topic Node

A Custom Topic Table

Topic	Term	Role	Weight
analytics	analytics	noun	1.0
analytics	analyze	verb	0.9
analytics	logistic regression	NOUN_GROUP	0.5
data	data	noun	1.0
data	data warehouse	NOUN_GROUP	0.7
data	analyze	verb	0.2

Columns in the custom topic table have names: **_topic_**, **_term_**, **_role_**, and **_weight_**.

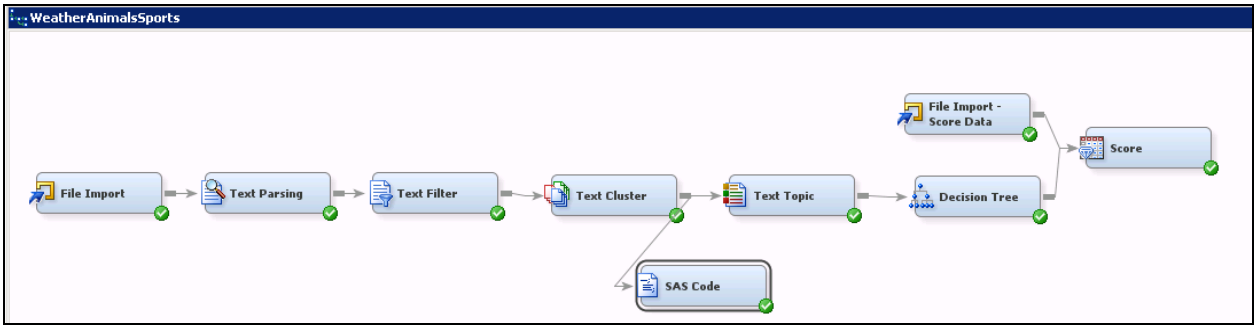
73

Weights can be any numeric value, positive or negative. Negative weights imply that the term supports the negative, or opposite, of the concept. A 0-1 system is the easiest to use.

Text Analytics Illustrated with a Simple Data Set

This demonstration illustrates some text analytic results using a simple data set that is designed to be easy to interpret. You can learn a lot about many features of the major Text Mining nodes by working through this example. You also use a SAS Code node to show you how to "get under the hood" and examine some results.

In this class, the project that you use and the diagrams are already set up, at least partially. However, for each demonstration, you should rebuild your own version of each diagram. In some cases, you can make additions to an existing diagram. You start by opening the project called **DMTXT13_1**. Then, select the diagram **WeatherAnimalsSports**. Set up the flow for this diagram as it is shown below.



1. Insert a **File Import** node in the diagram. This is the first node on the left that you see in the diagram above. In this demo, you use a data set that is completely stored in a single Excel spreadsheet. This is one way of getting relatively small text mining data sets into SAS Enterprise Miner. (In another chapter, how to use the Text Import node is discussed. It is run in a different way than the File Import node.) On the Property Sheet for the File Import node, specify the import file as the data set **D:\workshop\winsas\DMTXT13_1\WeatherAnimalsSports.xls**. Then run this node. Run the File Import node.

2. To see the data set after the **File Import** node is run, go to the **Exported Data** line of the Property Sheet. Click the ellipsis button (...). Then select the **Train** data and click **Browse** near the bottom of the window. You see the rows of the data set. The first seven rows are shown in the display below.

	Target_Subject	TextField
1	A	Bob has two dogs and one cat. The cat is bigger than either of the dogs.
2	S	Carmelo Anthony scored 42 points to lead the NY Knicks basketball team to a win over the Florida Pelicans.
3	S	Come play baseball with us.
4	S	Derek Jeter, the captain of the New York Yankees baseball team, said 2014 will be his last season playing.
5	S	Do you have a baseball or a football that we could play with? You can be on my team.
6	A	Do you like big dogs or little dogs? Dogs are such wonderful animals.
7	W	During the winter, the sun is lower in the sky than it is during the summer. That's why winter days are colder than summer days.

   The data set has two fields: **Target_Subject** (with values *A*, *S*, *W*) and **TextField**, which consists of short sentences. The sentences are about with one of three subjects: **Animals (A)**, **Sports (S)**, a **Weather (W)**.

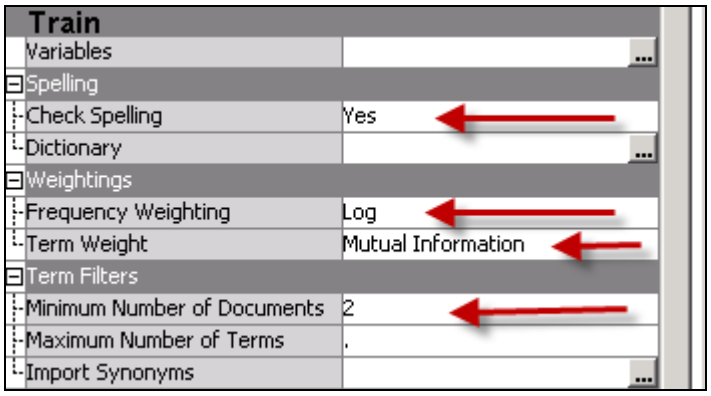
   It is important to understand that the **Target_Subject** field was created by a person interpreting the content of each **TextField**. It was not created automatically by the Text Miner nodes.

   Read through a few of the rows and make sure that you understand the nature of the data set and how it is structured. The variable **TextField** is what is referred to as a *document*. All the rows of **TextField** together (47 rows of data) are referred to as the *corpus collection*.

3. Attach a **Text Parsing** node to the **Text Import** node. This node has the language processing algorithms and has many different options that can be set by the user. For this demonstration, use the default settings. Run the **Text Parsing** node.

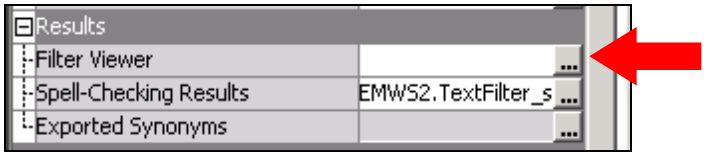
4. Attach a **Text Filter** node to the **Text Parsing** node. Change **Frequency Weighting** from **Default** to **Log**. Change **Term Weight** from **Default** to **Mutual Information**. Notice that **Mutual Information** is recommended for data where a target variable is present and predictive modeling is the goal. Also change the **Minimum Number of Documents** value in the Property Sheet to *2*. This option filters out terms that are not used in at least two documents in the corpus collection. Because you use a very small data set, this number is reduced from the default 4 to 2. Also, change the **Check Spelling** property from **No** to **Yes**. (It is easy to forget that **Check Spelling** is in the **Text Filter** node and not on the **Text Parsing** node. In general, changing this to **Yes** can add a lot of time to processing, so be cautious about its use.)

The settings for the **Text Filter** node now resemble the following:

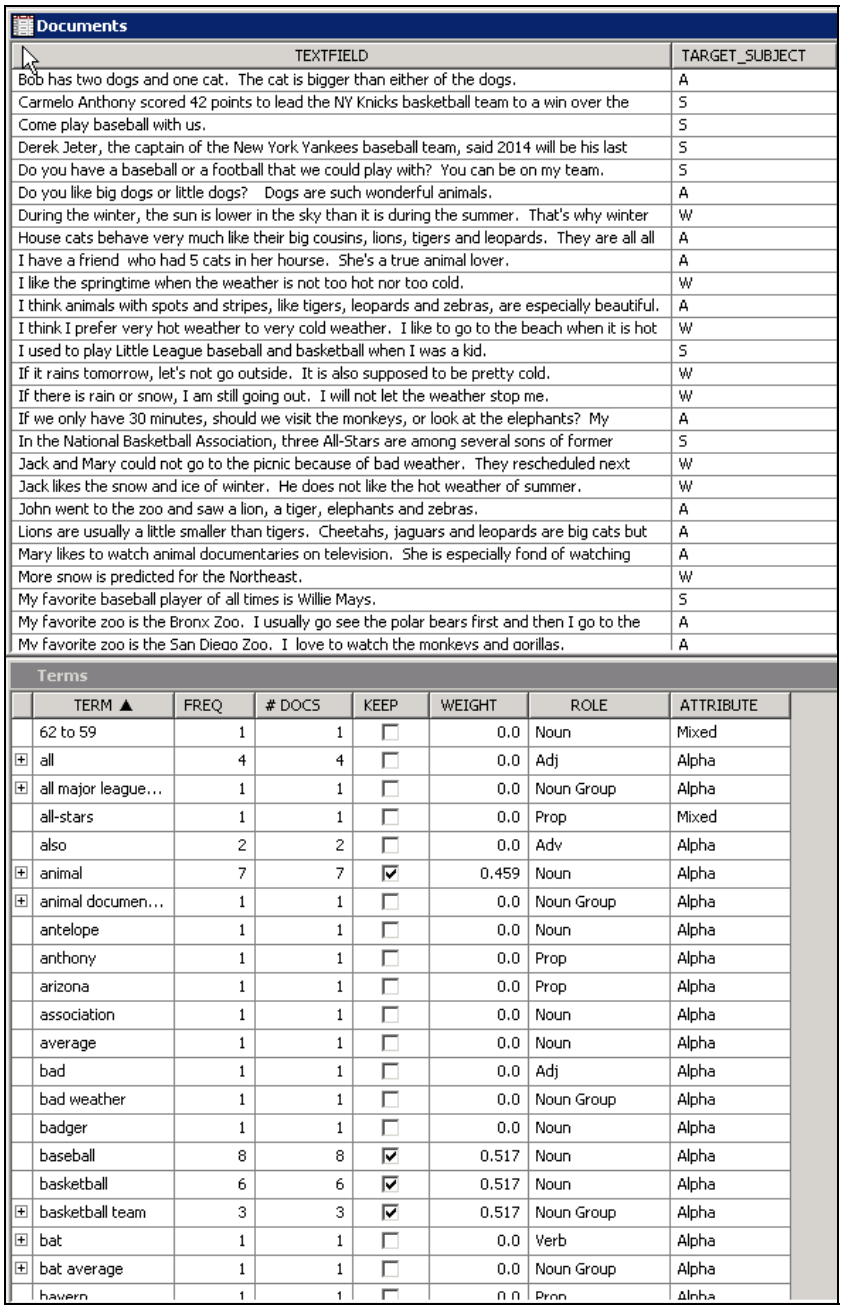


Run the **Text Filter** node.

5. Open the **Filter Viewer** in the Property Sheet. This is also called the *Interactive Filter Viewer*.



Look at the two main windows that open in the Filter Viewer. You see what is shown in the display below. The first window, labeled **Documents**, simply lists each document and any other variables on the data set; in this case, only the variable **Target_Subject**. The second window, labeled **Terms**, gives information about each of the terms that came out of the **Text Parsing** node. Notice that a *term* does not need to be a single word.



Documents

TEXTFIELD	TARGET_SUBJECT
Bob has two dogs and one cat. The cat is bigger than either of the dogs.	A
Carmelo Anthony scored 42 points to lead the NY Knicks basketball team to a win over the	S
Come play baseball with us.	S
Derek Jeter, the captain of the New York Yankees baseball team, said 2014 will be his last	S
Do you have a baseball or a football that we could play with? You can be on my team.	S
Do you like big dogs or little dogs? Dogs are such wonderful animals.	A
During the winter, the sun is lower in the sky than it is during the summer. That's why winter	W
House cats behave very much like their big cousins, lions, tigers and leopards. They are all all	A
I have a friend who had 5 cats in her hourse. She's a true animal lover.	A
I like the springtime when the weather is not too hot nor too cold.	W
I think animals with spots and stripes, like tigers, leopards and zebras, are especially beautiful.	A
I think I prefer very hot weather to very cold weather. I like to go to the beach when it is hot	W
I used to play Little League baseball and basketball when I was a kid.	S
If it rains tomorrow, let's not go outside. It is also supposed to be pretty cold.	W
If there is rain or snow, I am still going out. I will not let the weather stop me.	W
If we only have 30 minutes, should we visit the monkeys, or look at the elephants? My	A
In the National Basketball Association, three All-Stars are among several sons of former	S
Jack and Mary could not go to the picnic because of bad weather. They rescheduled next	W
Jack likes the snow and ice of winter. He does not like the hot weather of summer.	W
John went to the zoo and saw a lion, a tiger, elephants and zebras.	A
Lions are usually a little smaller than tigers. Cheetahs, jaguars and leopards are big cats but	A
Mary likes to watch animal documentaries on television. She is especially fond of watching	A
More snow is predicted for the Northeast.	W
My favorite baseball player of all times is Willie Mays.	S
My favorite zoo is the Bronx Zoo. I usually go see the polar bears first and then I go to the	A
My favorite zoo is the San Diego Zoo. I love to watch the monkeys and gorillas.	A

Terms

	TERM ▲	FREQ	# DOCS	KEEP	WEIGHT	ROLE	ATTRIBUTE
	62 to 59	1	1	☐	0.0	Noun	Mixed
⊞	all	4	4	☐	0.0	Adj	Alpha
⊞	all major league...	1	1	☐	0.0	Noun Group	Alpha
	all-stars	1	1	☐	0.0	Prop	Mixed
	also	2	2	☐	0.0	Adv	Alpha
⊞	animal	7	7	☑	0.459	Noun	Alpha
⊞	animal documen...	1	1	☐	0.0	Noun Group	Alpha
	antelope	1	1	☐	0.0	Noun	Alpha
	anthony	1	1	☐	0.0	Prop	Alpha
	arizona	1	1	☐	0.0	Prop	Alpha
	association	1	1	☐	0.0	Noun	Alpha
	average	1	1	☐	0.0	Noun	Alpha
	bad	1	1	☐	0.0	Adj	Alpha
	bad weather	1	1	☐	0.0	Noun Group	Alpha
	badger	1	1	☐	0.0	Noun	Alpha
	baseball	8	8	☑	0.517	Noun	Alpha
	basketball	6	6	☑	0.517	Noun	Alpha
⊞	basketball team	3	3	☑	0.517	Noun Group	Alpha
⊞	bat	1	1	☐	0.0	Verb	Alpha
⊞	bat average	1	1	☐	0.0	Noun Group	Alpha
	bavern	1	1	☐	0.0	Prop	Alpha

The information shown in the Terms window is the following:

FREQ = number of times the term appeared in the entire corpus

#DOCS = total number of documents in which the term appeared

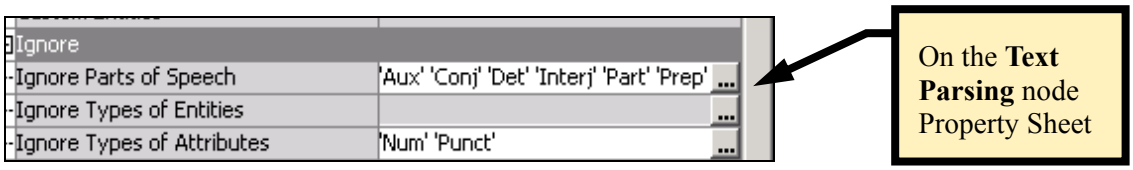
KEEP = whether the term is kept for calculations

WEIGHT = a term weight (in this case, Mutual Information, which is discussed in a later chapter) because you specified **entropy** as the term weight to use in the Property Sheet

ROLE = part of speech of the term

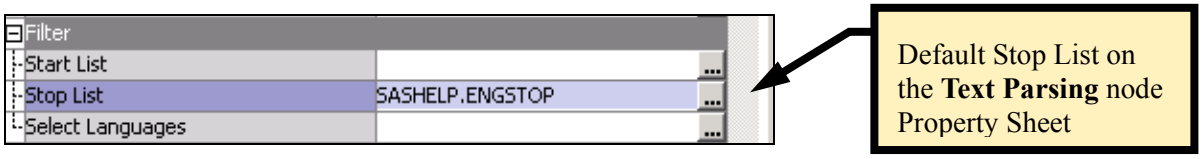
ATTRIBUTE = the different categories are listed at the end of this chapter

Go to the **Terms** window and confirm that "the" is not listed. (If the column of **Terms** is not already in alphabetical order, you can sort a column by clicking on the heading.) Why does the most common word in the English language not appear on the list? To understand why, click the **Text Parsing** node so that the Property Sheet for that node is visible. Look at the properties near the bottom. You can see that there is an **Ignore Parts of Speech** property. By default, this excludes certain terms that are very common. In particular, *Det* represents *Determiner*, which is a class of common words and phrases such as *the*, *that*, *an*, and so on. These are eliminated unless you modify this property.

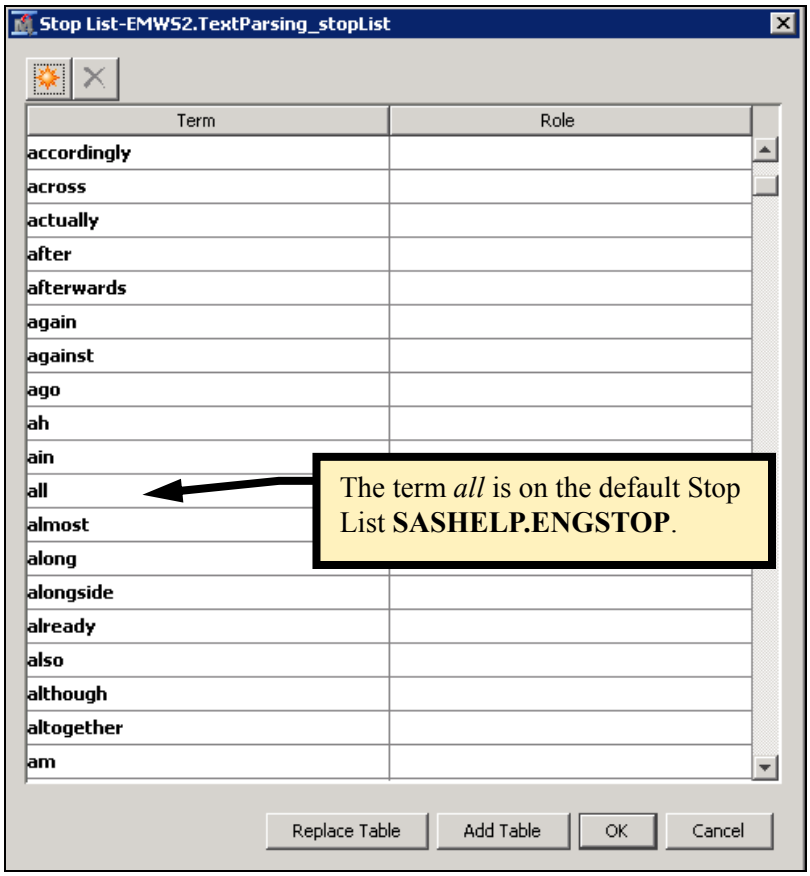


Go back to the **Text Filter** node. Why are some of the terms kept (KEEP is checked), but others are not kept (KEEP is unchecked)? There are several reasons why a word is not kept, and these can depend on settings in both the Text Parsing node and the Text Filter node. One reason, such as for the word *antelope*, is that it does not appear in enough documents. You previously set the Minimum Number of Documents property to **2** for the Text Filte**r** node. Because *antelope* occurs only in one document, it is not kept.

Another reason a term is not kept is if it appears on a Stop List used in the Text Parsing node. The default Stop List is **SASHELP.ENGSTOP**. If you open and look at it, you see a list of many terms that are excluded from further computations.



If you open **SASHELP.ENGSTOP** from the Text Parsing node, you see that *all* is listed as a term not to be used, as in the display below. Therefore, *all* is not selected as **KEEP** in the Text Filter node.



6. You now use the two main analytic text mining tools, the **Text Clustering** node and the **Text Topic** node. Attach a **Text Clustering** node to the **Text Filter** node as in the first diagram of this section. The Text Clustering node takes the 47 documents in the example data set and separates them into *mutually exclusive* and *exhaustive* groups (that is, clusters). The number of clusters to be used is under user control. You modify four of the default settings.

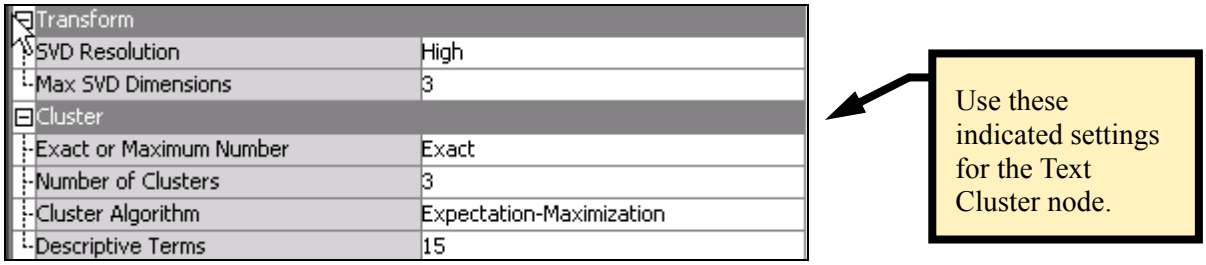
   a. Change **SVD Resolution** from *Low* to **High**.

   b. Change **Max SVD Dimensions** from *100* to **3**.

   c. Change **Exact or Maximum Number** (of clusters) to **Exact**.

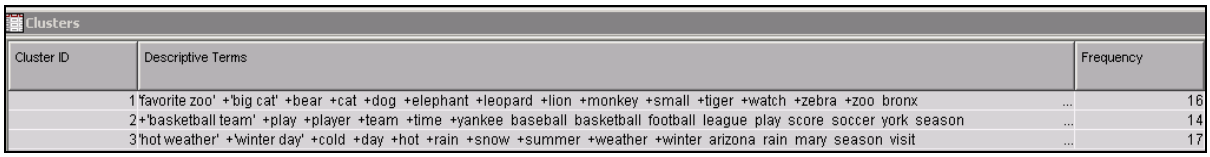
   d. Change **Number of Clusters** from *40* to **3**.

   The settings resemble the ones below.



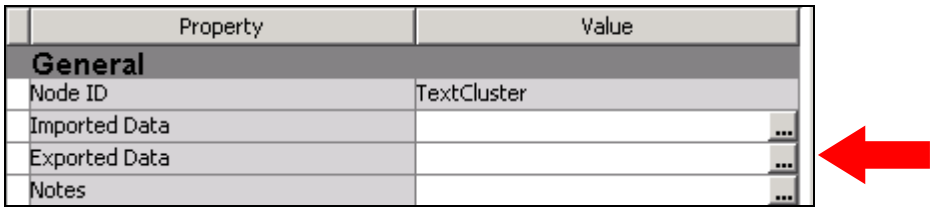
Regarding the **Text Cluster** properties, remember that you are using a very small and simple data set. You know that there are basically three types of documents (animals, sports, weather). It is most reasonable to think in terms of creating a small number of clusters (for example, three to five). Use ***three***. In practice, with real and complex text data, you want to experiment with these parameters. You might want to start with the default property settings. Run the node.

7. Open the **Text Cluster** node results and examine the left side of the Clusters window as shown.

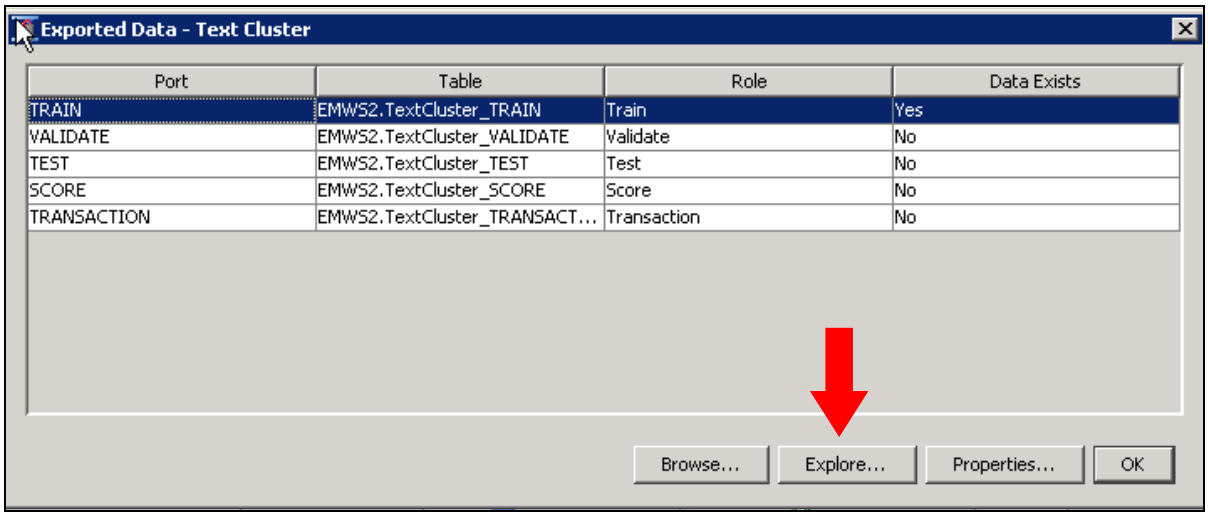


Exactly three clusters were created, as requested in the Property Sheet. The **Descriptive Terms** column shows up to 15 terms that are given to help the user interpret the types of documents that are put into each cluster. (The number can be changed.) These terms are selected by the underlying algorithm as being the most important for characterizing the documents placed into a given cluster. Reading these, you can see that Cluster 1, which has 16 documents, has terms such as *favorite zoo*, *big cat*, and so on. These documents are likely about animals. The + indicates a stemmed term. Cluster 2 has 14 documents that are likely related to sports. Cluster 3 has 17 documents that likely deal with weather.

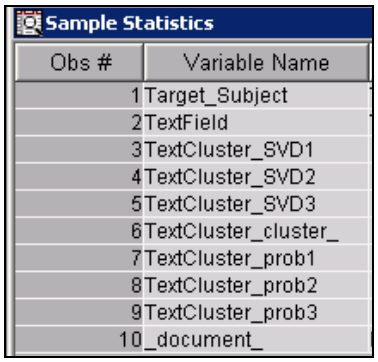
8. To see the new variables that were generated by the Text Cluster node, close out of the results. Select **Exported Data** from the Property Sheet.



Then select the **Train** data set and click **Explore**.



The upper right window (Sample Statistics) shows a list of variables that were exported from the Text Cluster node.


Sample Statistics

Obs #	Variable Name
1	Target_Subject
2	TextField
3	TextCluster_SVD1
4	TextCluster_SVD2
5	TextCluster_SVD3
6	TextCluster_cluster_
7	TextCluster_prob1
8	TextCluster_prob2
9	TextCluster_prob3
10	_document_

Several new variables have been added to the original variables **Target_Subject** and **TextField**:

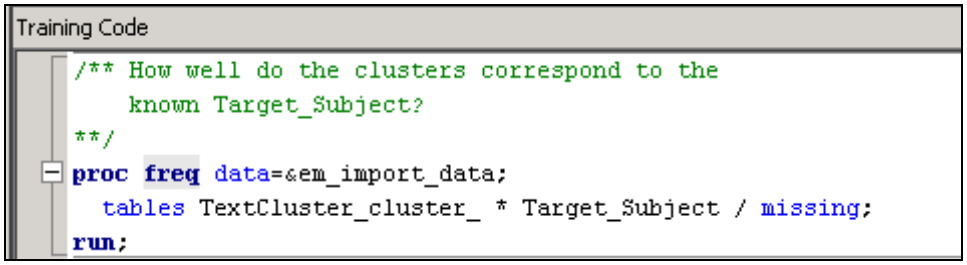
TextCluster_SVD1-TextCluster_SVD3 – These are numeric variables calculated from a singular value decomposition of the (usually weighted) document-term frequency matrix. Each document is represented by its values on these three new variables. The values are also normalized so that for each document all the squared SVD values sum to 1.0. These are the variables that are used to cluster the documents. (Discussion of the calculation of the SVD values is in Chapter 3.)

TextCluster_cluster_ – This is the Cluster ID, a categorical variable. In this example, it is simply a number from 1 to 3 because three clusters were created. The clusters were generated by performing a cluster analysis on the three **TextCluster_SVD** variables. The interpretation of the clusters begins with looking at the descriptive terms given for each cluster, as you did earlier.

TextCluster_prob1 - TextCluster_prob3 – These variables are the probabilities of membership in each cluster for a given document. The sum of these probabilities is 1. A document is assigned to the cluster where it has the highest membership probability.

document – This is a document ID.

9. Many ways to do further explorations with these results can be helpful for learning about what the text mining nodes are doing and for looking more deeply at certain aspects of the analysis. SAS Enterprise Miner provides a SAS Code utility node that is especially good for this. Attach a **SAS Code** node to the **Text Cluster** node and go into the Code Editor on the Property Sheet. Enter the following code:


Training Code

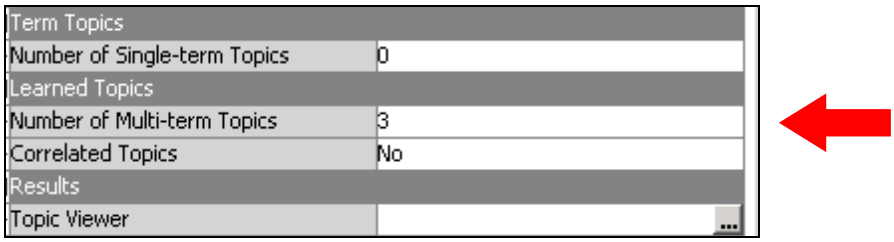
```
/** How well do the clusters correspond to the
    known Target_Subject?
**/
proc freq data=&em_import_data;
  tables TextCluster_cluster_ * Target_Subject / missing;
run;
```

The macro variable **&em_import_data** refers to the Training data set ***after*** it is processed by the Text Cluster node and imported into the SAS Code node. Because there is a target variable (**Target_Subject**) created by a person who read the documents, it is interesting to see how the clusters automatically created by the Text Cluster node align with how the documents were labeled by the person. The PROC FREQ step does this by cross-tabulating the cluster variable (**TextCluster_cluster_**) with the target. Run this code and look at the results.

```
The FREQ Procedure

Table of TextCluster_cluster_ by Target_Subject

TextCluster_cluster_
          Target_Subject(Target_Subject)

Frequency|
Percent  |
Row Pct  |
Col Pct  |A       |S       |W       |  Total
         |        |        |        |
         |        |        |        |
---------+--------+--------+--------+
       1 |     16 |      0 |      0 |     16
         |  34.04 |   0.00 |   0.00 |  34.04
         | 100.00 |   0.00 |   0.00 |
         |  94.12 |   0.00 |   0.00 |
---------+--------+--------+--------+
       2 |      0 |     14 |      0 |     14
         |   0.00 |  29.79 |   0.00 |  29.79
         |   0.00 | 100.00 |   0.00 |
         |   0.00 | 100.00 |   0.00 |
---------+--------+--------+--------+
       3 |      1 |      0 |     16 |     17
         |   2.13 |   0.00 |  34.04 |  36.17
         |   5.88 |   0.00 |  94.12 |
         |   5.88 |   0.00 | 100.00 |
---------+--------+--------+--------+
Total          17       14       16       47
             36.17    29.79    34.04   100.00
```

This crosstabulation shows that Cluster 1 (which was seen previously to have descriptive terms such as *favorite zoo*, *big cat*, and so on) consists of 16 documents defined that have to do with animals (A) as labeled by the human reader. Cluster 2 (*basketball team*, *play*, and so on) consists of 14 documents with a target value always equal to S. Cluster 3 (*hot weather*, *winter day*, and so on) consists of 17 documents, and 16 of them were defined as weather-related. The three clusters line up almost perfectly with the labels given to the documents. ***It would be wonderful if real data worked out this well, but do not expect that!***

10. Set up and run the Text Topic node. Look at some results to see how they compare with the Text Cluster node results. Although a cluster is a mutually exclusive category (that is, each document can belong to one and only one cluster), a document can have more than one topic or it can have none of the topics. Attach a **Text Topic** node directly to the **Text Cluster** node. Make one change to the default properties by specifying **3** as the number of multi-term topics to create. Just as the number of clusters created is a parameter with which you want to experiment when you use the Text Cluster node, this parameter for the number of topics to create is typically something that you might try with different values. In this example, the artificial data set was purposely created with three different topics, so a reasonable value to start with would be 3 to 5 and not the default value of 25. You use 3.



Run the node. Then click on the ellipsis for the **Topic Viewer** on the Property Sheet. The Topic Viewer is an interactive group of windows. The Topics window shows the topics created by the node.

Topics

Topic	Category	Term Cutoff	Document Cutoff	Number of Terms	# Docs
+snow,+cold,+winter,+weather,+hot	Multiple	0.189	0.469	7	9
baseball,+team,basketball,+play,+basketball team	Multiple	0.192	0.351	5	8
+lion,+tiger,+zoo,+animal,+zebra	Multiple	0.196	0.363	7	8

The three topics created by the algorithm also have key descriptive terms to guide interpretation. The five most descriptive terms for each topic are shown. By default, the first topic is selected when you open the viewer. In this example, its descriptive terms start with *snow, cold, ….* This is evidently a topic related to weather. The second topic has descriptive terms starting with *baseball, team, …* and relating to sports. The descriptive terms for the third topic (*lion, tiger, …*) are interpretable as having to do with animals. With this simple data set, the algorithm did very well in identifying what are known to be the three underlying topics in the documents.

In the Topics window, there is a column labeled **Term Cutoff**. For each created topic, the algorithm computes a topic weight for every term in the corpus. This measures how strongly the term represents or is associated with the given topic. Terms that are above a certain value, called the *Term Cutoff*, appear in yellow in the Terms window shown below.

Look at the Terms window. You can see all the terms above the cutoff value.

Terms

Topic Weight	+	Term	Role	# Docs	Freq
0.496	+	snow	Noun	8	10
0.487	+	cold	Adj	9	9
0.385	+	winter	Noun	6	7
0.338	+	weather	Noun	7	8
0.288	+	hot	Adj	6	7
0.247	+	summer	Noun	4	5
0.243	+	day	Noun	5	6
0.135		rain	Noun	3	3
0.102	+	rain	Verb	2	2
0.095	+	winter day	Noun Group	2	2
0.089		hot weather	Noun Group	3	3
0.04	+	animal	Noun	8	8

You should know, however, that ***all*** terms have a topic weight for each topic, although it might be a very small value.

Part of the Documents window at the bottom of the Interactive Viewer is show below. Every document receives a topic weight for each topic. (This is discussed in Chapter 3.)

Documents

Topic Weight	TextField	Target_Subject
0.959	The weather in NYC is hot in the summer and cold in the winter, but we do not get as much snow as in	W
0.774	Jack likes the snow and ice of winter. He does not like the hot weather of summer.	W
0.704	Winter is my favorite season. I love the cold and the snow.	W
0.682	This has been a very difficult winter, much colder than usual with lots of snow, ice and rain.	W
0.677	During the winter, the sun is lower in the sky than it is during the summer. That's why winter days are	W
0.612	I think I prefer very hot weather to very cold weather. I like to go to the beach when it is hot and sunny.	W
0.599	I like the springtime when the weather is not too hot nor too cold.	W
0.569	We have had snow, snow and more snow for 10 days in a row.	W
0.556	Winter days are often so pretty, even if it is cold.	W
0.446	If there is rain or snow, I am still going out. I will not let the weather stop me.	W
0.41	Phoenix, Arizona, had it's fourth hottest day on record in June, 2013, when the temperature reached 119	W
0.38	More snow is predicted for the Northeast.	W

Notice that in the Documents window, the documents with topic weight values above the document cutoff for this topic (.469) are shown in yellow. However, it is important to observe that there are several documents ***below*** this cutoff value that are nevertheless related to a weather topic. For example, the first document below the cutoff ("If there is rain or snow....") has a topic weight equal to .446 and is not highlighted in yellow. However, this document certainly involves weather. Although a cutoff value for a document can be useful in helping to understand what the topic represents, for some purposes, it is the topic weight itself that is used, such as in predictive modeling. It is also possible to change the cutoff.

To see what variables were generated by the Text Topic node, as was done previously with the Text Cluster node, go to **Exported Data** in the Property Sheet. Select the **Train** data set and click **Explore**. The list of all the variables shows what was previously created by the Text Cluster node and the new **TextTopic** and **TextTopic_raw** variables created by the Text Topic node.

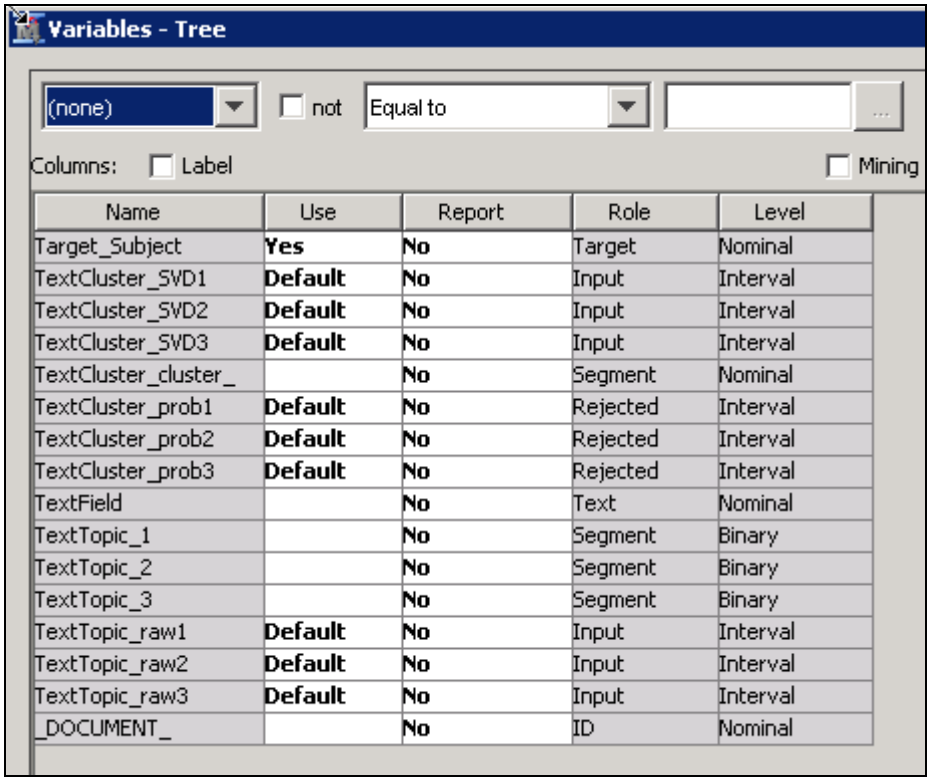
Sample Statistics

Obs #	Variable Name	Label
1	Target_Subject	Target_Subject
2	TextField	TextField
3	TextCluster_SVD1	
4	TextCluster_SVD2	
5	TextCluster_SVD3	
6	TextCluster_cluster_	
7	TextCluster_prob1	
8	TextCluster_prob2	
9	TextCluster_prob3	
10	TextTopic_1	_1_0_+snow,+cold,+winter,+weather,+hot
11	TextTopic_2	_1_0_baseball,+team,basketball,+play,+basketball team
12	TextTopic_3	_1_0_+lion,+tiger,+zoo,+animal,+zebra
13	TextTopic_raw1	+snow,+cold,+winter,+weather,+hot
14	TextTopic_raw2	baseball,+team,basketball,+play,+basketball team
15	TextTopic_raw3	+lion,+tiger,+zoo,+animal,+zebra
16	_DOCUMENT_	Document

TextTopic_raw1 - TextTopic_raw3 – These are numeric variables that indicate the strength a particular topic has within a given document. Three topics were generated because this was specified on the Property Sheet. These variables are the same as the topic weight values for the documents that were previously looked at in the Documents window of the interactive Topic Viewer. Each of these variables (topics) has a label (the five most descriptive terms) to identify it and help the user interpret the topic.

TextTopic_1 - TextTopic_3 – These are ***binary*** variables defined for each document and constructed from the **TextTopic_raw1 - TextTopic_raw3** values based on the document cutoff values described earlier. For example, **TextTopic_1** is set to *1* if a document has a **TextTopic_raw1** value greater than the cutoff value for this particular topic. Otherwise, it is set to 0. The labels for the **TextTopic** variables are the same as for the **TextTopic_raw** raw variables, except that they have _1_0_ as prefixes. This indicates that they are binary variables. Each label shows the five most descriptive terms that are identified with that topic.

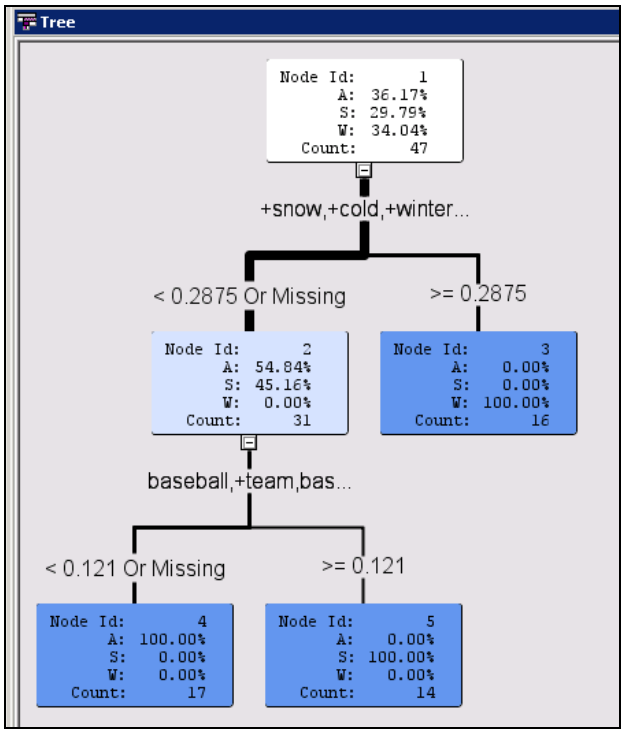
11. The emphasis in this class is on text mining for prediction (including supervised classification). To that end, continue this demonstration by attaching a **Decision Tree** node to the output of the **Text Topic** node. This node is found on the Model tab at the top. Among all model types, decision tree models are especially good for interpretation. After you attach the Decision Tree node but before running it with the default settings, go to the **Variables** ellipsis button in the Property Sheet to view the following window:



Name	Use	Report	Role	Level
Target_Subject	**Yes**	**No**	Target	Nominal
TextCluster_SVD1	**Default**	**No**	Input	Interval
TextCluster_SVD2	**Default**	**No**	Input	Interval
TextCluster_SVD3	**Default**	**No**	Input	Interval
TextCluster_cluster_		**No**	Segment	Nominal
TextCluster_prob1	**Default**	**No**	Rejected	Interval
TextCluster_prob2	**Default**	**No**	Rejected	Interval
TextCluster_prob3	**Default**	**No**	Rejected	Interval
TextField		**No**	Text	Nominal
TextTopic_1		**No**	Segment	Binary
TextTopic_2		**No**	Segment	Binary
TextTopic_3		**No**	Segment	Binary
TextTopic_raw1	**Default**	**No**	Input	Interval
TextTopic_raw2	**Default**	**No**	Input	Interval
TextTopic_raw3	**Default**	**No**	Input	Interval
DOCUMENT		**No**	ID	Nominal

By default, the only text mining variables that are considered as candidate prediction variables are those that have a role of **Input**. These are **TextCluster_SVD** and **TextTopic_raw** variables. Others, such as the **TextCluster_cluster_** or the **TextTopic** variables, which some analysts would consider using for prediction or classification, must be redefined as Input variables using the Metadata node.

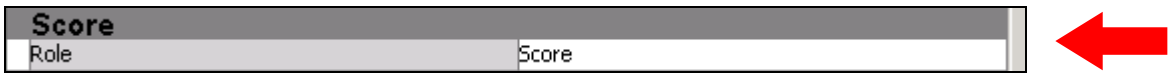
12. Run the default Decision Tree node. Open the results and maximize the Tree window.



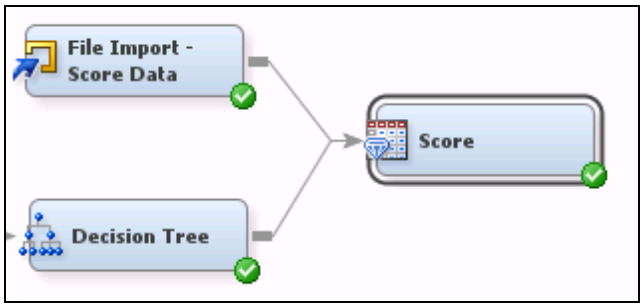
The decision tree resulted in three leaves. They are 100% accurate in classifying the documents as either A, S, or W. The variables used for this prediction or classification are the **TextTopic_1** and **TextTopic_2** text mining variables because the labels for these variables are displayed in the tree. The rules for the tree are obvious. The leaf with Node Id=3 comprises all documents where **TextTopic_1** (+snow, +cold, + winter …) is quite high, that is, >=.2875. Then Node 4 consists of documents where **TextTopic_1** is less than .2875 and **TextTopic_2** (baseball,+team, …) is less than .121. That is, Node 4, which has 100% animal documents, consists of documents that do not contain much information about either weather or sports. Finally, Node 5 is defined as consisting of documents that have **TextTopic_1** less than .2875 and **TextTopic_2** greater than .121. In other words, these are documents relatively high on the sports topic and low on the weather topic.

There are many approaches that an analyst can use to interpret the results of text mining. In this demonstration, the situation is easy to understand. In most realistic applications, you might need to do some creative analytic work to dig more deeply. (Some ideas for this are presented in later chapters.)

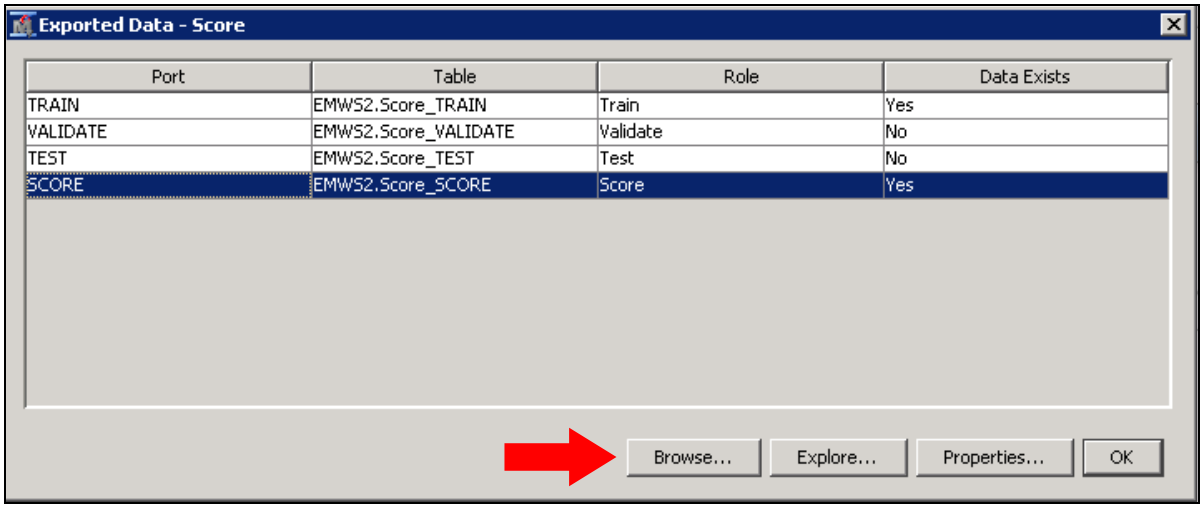
13. The final part of this demonstration is to use the Score node to score new data set. Following the top part of the diagram shown at the beginning of this demonstration, bring in a new **File Import** node. Rename it **File Import Score Data**. The import file for scoring is **D:\workshop\winsas\dmtxt13_1\Score_WeatherAnimalsSports.xls**. In the Property Sheet, change the role of the data set to **Score**.



Run the node and look at the Exported Data window. This **Score** data set has 16 documents. They are related to the three subjects (animals, sports, or weather). (As is usually the case with a data set to be scored, there is no target field on this data set.) The object now is to classify these documents using the Decision Tree model that was previously obtained on the training data, **WeatherAnimalsSports.xls**. To do that, bring in a **Score** node and connect it to the output of the **File Import Score Data** node and also to the output of the **Decision Tree** node.



Run the **Score** node. Then go to the Exported Data window through the Property Sheet. Select the **SCORE** data to view and click **Browse**.



14. When the Browse window appears, move. the column headings so that **TextField** is the first column heading and **Into: Target_Subject** is the second heading (See the display below.) Into: Target_Subject is the label for the variable **I_Target_Subject**. This variable is the predicted classification (A,S, or W) of **Target_Subject** based on the **TextTopic_1** and **TextTopic_2** variables used in the decision tree.

    Read through the 16 rows and check to see whether any of the classifications looks incorrect to you. Generally, all of them should look right except observation #14, which is about seasons and therefore is really a document about weather. However, it was incorrectly classified as A, an animal document. (Also, observation #16 is probably a better fit in A, but does fit in W, as it was used here.)

EMWS2.Score_SCORE

	TextField	Into: Target_Subject
1	We have a dog in our house. His name is Princey and he is a part of our family.	A
2	I spend a lot of time on the weekend watching sports shows: football, baseball, basketball and soccer are all fun for me.	S
3	The World Cup in soccer is held every four years. Brazil has had the winning team more than any other country.	S
4	The 2013 World Series was won by the Boston Red Sox team. They beat the St. Lous Cardinals in 6 games.	S
5	The winter weather has been very harsh in many parts of the U.S. during the months of January and February.	W
6	Yesterday, I watched a documentary about the big cats of Africa.	A
7	In our neighborhood, one man has 5 small dogs that he walks every day.	A
8	We have a problem with feral cats.	A
9	Professional basketball players are paid enormous salaries.	S
10	We have friends who live in a rural area and they sometimes see bears near their house.	A
11	We have had 5 inches of snow since this morning.	W
12	Rainy weather in the summer can be very pleasant to cool things off.	W
13	I could watch elephants all day. They are such beautiful, graceful animals.	A
14	I would like to spend summers in Maine and winters in Florida.	A
15	Leopards are nocturnal hunters.	A
16	I watched a nature movie about bears hibernating in winter and waking up in the springtime.	W

There is an endless number of reasons why the underlying text mining and modeling algorithms make mistakes. One possibility in this case is the very small number of training examples that were used.

1.01 Multiple Choice Poll

When you run the Text Cluster node, by default, which of the following variables are given the role of input for a predictive model?

a. the **TextCluster_SVD** variables
b. the **TextCluster_cluster_** variable
c. the **TextCluster_prob** variables
d. b and c above

75

Technical Details

The following material is extracted from the Reference Help for SAS Enterprise Miner.

Parts of Speech in SAS Text Miner

SAS Text Miner can identify the part of speech for each term in a document based on the context of that term. Terms are identified as one of the following parts of speech:

- Abbr (abbreviation)
- Adj (adjective)
- Adv (adverb)
- Aux (auxiliary or modal)
- Conj (conjunction)
- Det (determiner)
- Interj (interjection)
- Noun (noun)
- Num (number or numeric expression)
- Part (infinitive marker, negative participle, or possessive marker)
- Pref (prefix)
- Prep (preposition)
- Pron (pronoun)
- Prop (proper noun)
- Punct (punctuation)
- Verb (verb)
- VerbAdj (verb adjective)

Noun Groups in SAS Text Miner

SAS Text Miner can identify noun groups, such as *clinical trial* and *data set*, in a document collection. Noun groups are identified based on linguistic relationships that exist within sentences. Syntactically, these noun groups act as single units. Therefore, you can choose to parse them as single terms.

- If stemming is on, noun groups are stemmed. For example, the text *amount of defects* is parsed as *amount of defect*.
- Frequently, shorter noun groups are contained within larger noun groups; both the shorter and larger noun groups appear in parsing results.

Entities in SAS Text Miner

An *entity* is any of several types of information that SAS Text Miner can distinguish from general text. If you enable SAS Text Miner to identify them, entities are analyzed as a unit, and they are sometimes normalized. When SAS Text Miner extracts entities that consist of two or more words, the individual words of the entity are also used in the analysis.

Out of the box, SAS Text Miner identifies the following standard entities:

- ADDRESS (postal address or number and street name)
- COMPANY (company name)
- CURRENCY (currency or currency expression)
- DATE (date, day, month, or year)
- INTERNET (e-mail address or URL)
- LOCATION (city, country, state, geographical place/region, political place/region)
- MEASURE (measurement or measurement expression)
- ORGANIZATION (government, legal, or service agency)
- PERCENT (percentage or percentage expression)
- PERSON (person's name)
- PHONE (phone number)
- PROP_MISC (proper noun with an ambiguous classification)
- SSN (Social Security number)
- TIME (time or time expression)
- TIME_PERIOD (measure of time expressions)
- TITLE (person's title or position)
- VEHICLE (motor vehicle including color, year, make, and model)

You can also use SAS Content Categorization with Teragram Contextual Extraction to define custom entities and import these for use in a Text Parsing node. When you create compiled custom entity files, ensure that you specify September 14, 2009 as the compatibility date. (Valid files have the extension .li.) Otherwise, the files cannot be used in SAS Text Miner.

Entities are normalized in the following situations:

- SAS Text Miner uses a fixed dictionary of company and organization names in order to identify these entity types, and entities of this type are frequently associated with a parent. For example, if *IBM* appears in the text, it is returned with the predefined parent *International Business Machines*. Typically, the longest and most precise version of a name is used as the parent form.
- SAS Text Miner normalizes entities that have an ISO (International Standards Organization) standard (dates/years, currencies, and percentages). Rather than return the normalization as a parent of the original term, these normalizations actually replace the original term.
- You can alter any parent forms that are returned by editing the synonym list. Place terms that you want to identify as an entity in the *term* variable, the parent to associate with it in the *parent* variable, and place the entity category in the *category* variable. Then rerun the node.

Attributes in SAS Text Miner

When a document collection is parsed, SAS Text Miner categorizes each term as one of the following attributes, which gives an indication of the characters that compose that term:

- Alpha, if characters are all letters
- Num, if term characters include a number
- Punct, if the term is a punctuation character
- Mixed, if term characters include a mix of letters, punctuation, and white space
- Entity, if the term is an entity

Exercises

1. Changing Properties from the Demonstration to See How This Affects Results

Experiment with changing some of the properties from the setup in the demonstration. For example, you might retain all the previous settings but make the following changes:

- Change the Text Filter node Term Weight property to **Entropy** (instead of Mutual Information).
- Change the Text Cluster node Max SVD Dimensions and the Number of Clusters to **4** (instead of 3).
- Change the number of multi-term topics on the Text Topic node to **4** (instead of 3).

Look at all the results from the Text Topic and Text Cluster nodes and interpret them as you did in class. (If you copy the previous SAS Code node connected to Text Cluster, it does not run until you change the code in the PROC FREQ statement to use **TextCluster2_cluster_** instead of TextCluster_cluster_ because this is the ***second*** Text Cluster node used in the diagram.)

One point of this exercise is that changing some of the property settings on the nodes leads to different ways of looking at your data. You want to be experimental and creative in finding a good combination of interpretable results and predictive power.

1.4 Chapter Summary

Data mining incorporates analytic techniques applied to problems of pattern discovery and predictive modeling. SAS Enterprise Miner uses the SEMMA methodology for addressing data mining problems.

Text mining is a specialized area of data mining that brings together algorithms and methods from natural language processing and information retrieval to solve problems involving a collection of documents. SAS Text Miner provides modern techniques for solving text mining problems.

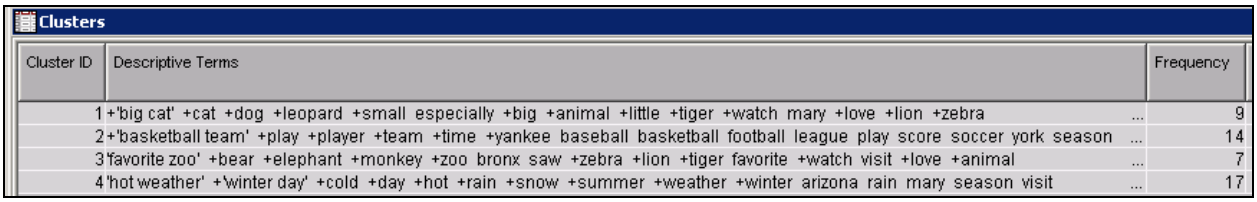
In this chapter, you used the following text mining nodes:

- Text Parsing
- Text Filter
- Text Cluster
- Text Topic

1.5 Solutions

Solutions to Exercises

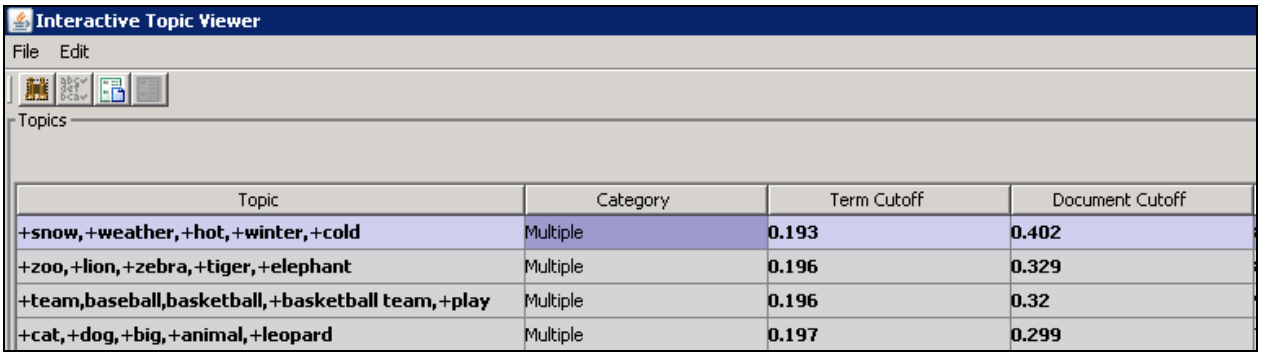
If you use SAS Enterprise Miner 13.1 and you ran the exercise with the specified property settings, the following Text Cluster and Text Topic results are obtained:


Clusters

Cluster ID	Descriptive Terms	Frequency
1	+'big cat' +cat +dog +leopard +small especially +big +animal +little +tiger +watch mary +love +lion +zebra ...	9
2	+'basketball team' +play +player +team +time +yankee baseball basketball football league play score soccer york season ...	14
3	'favorite zoo' +bear +elephant +monkey +zoo bronx saw +zebra +lion +tiger favorite +watch visit +love +animal ...	7
4	'hot weather' +'winter day' +cold +day +hot +rain +snow +summer +weather +winter arizona rain mary season visit ...	17

Notice that with four clusters, there are now two animal clusters (Cluster 1 and Cluster 3) instead of one.

Similarly, with four topics, there now are two (Topic 2 and Topic 4) that involve animals.

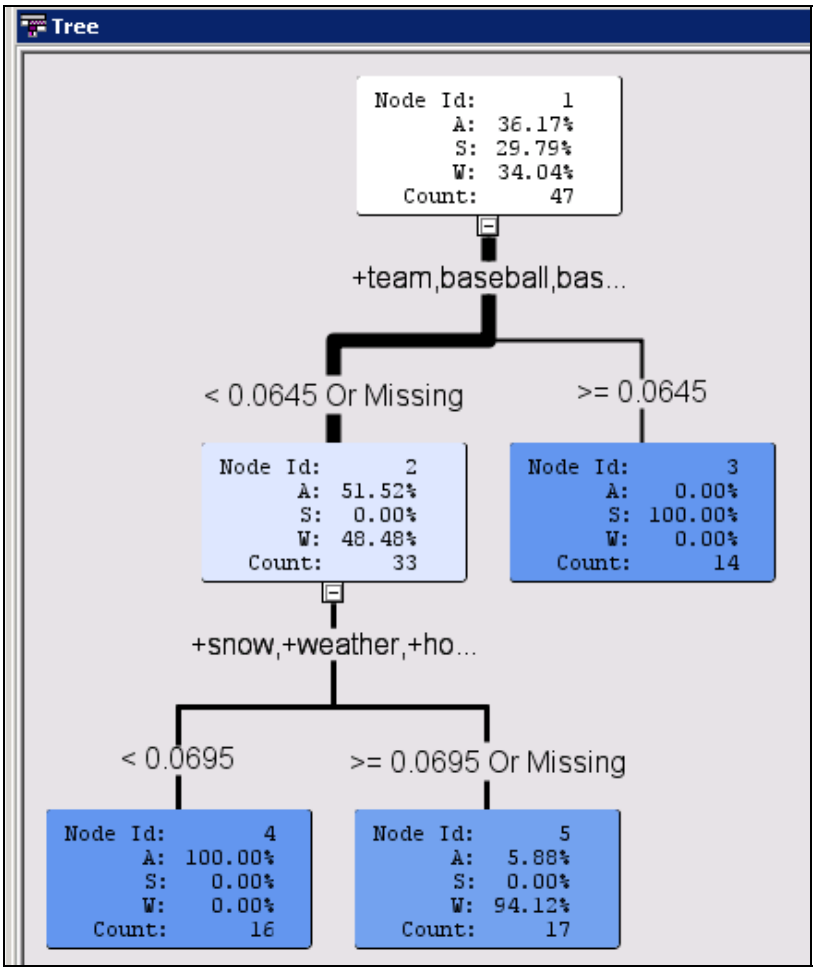

Interactive Topic Viewer

File Edit

Topics

Topic	Category	Term Cutoff	Document Cutoff
+snow,+weather,+hot,+winter,+cold	Multiple	0.193	0.402
+zoo,+lion,+zebra,+tiger,+elephant	Multiple	0.196	0.329
+team,baseball,basketball,+basketball team,+play	Multiple	0.196	0.32
+cat,+dog,+big,+animal,+leopard	Multiple	0.197	0.299

The decision tree resulting from these settings is shown below and indicates one misclassification on the training data.



Did you try some other property settings that give good results?

Solutions to Student Activities (Polls/Quizzes)

1.01 Multiple Choice Poll – Correct Answer

When you run the Text Cluster node, by default, which of the following variables are given the role of input for a predictive model?

a. the **TextCluster_SVD** variables
b. the **TextCluster_cluster_** variable
c. the **TextCluster_prob** variables
d. b and c above

76

Chapter 2 Overview of Text Analytics

2.1 Using the Text Import Node, Adding a Target Variable, and Comparing Models

Objectives

- Describe how the Text Import node is used for processing document collections and creating a single SAS data set for text mining.
- Show how the SAS data set created from Text Import can then be merged with another SAS data set containing target information and other non-text variables.
- Show how to compare two models, one using only conventional input variables and another using the conventional inputs ***and*** some text mining variables.

3

Often the most challenging part of the data mining process is obtaining and preprocessing the data. SAS provides a rich set of tools for data preparation.

SAS Data Access Features

- SAS provides ***access engines*** for commercial databases and common file types.
- SAS Enterprise Guide, the SAS windowing environment, and other SAS products or components support a ***Data Import Wizard***.
- The SAS language supports high-level and low-level ***I/O functions*** for reading data files.

4

SAS/ACCESS engines provide direct connectivity to popular commercial databases. A SAS/ACCESS engine hooks into the database supervisor to enable direct access to tables in the database. SAS/ACCESS engines also provide connectivity to common file formats, such as Microsoft Excel files.

The Data Import Wizard enables access to common file formats, including comma-separated values (CSV) and Microsoft Excel files.

The SAS language provides a complete set of data access functions, including functions for low-level file I/O, so that, theoretically, any file format can be read and processed.

2.01 Multiple Answer Poll

Select all data file formats that you routinely encounter in your work.

a. PC file formats, for example, Microsoft Excel (ODBC, OLE DB)
b. Microsoft SQL Server tables
c. Oracle tables
d. Sybase tables
e. Teradata tables
f. DB2 tables
g. MySQL tables
h. Informix tables
i. other

5

The SAS language also supports Perl regular expressions.

Selected Functionality of the SAS Language

- Very often your text files require considerable processing before they can be used.
- The SAS language provides numerous tools for doing this:
  - support for Perl regular expressions
  - character string functions for searching and modifying text data
  - mathematical and statistical functions for working with numeric data
  - formats and informats for reading and writing data in most recognized data formats
  - a macro facility to enable users to program complex operations for enterprise-wide use

6

Perl regular expressions enable you to use terse scripts for complex data operations on text files. For example, to preserve confidentiality, you might want to convert all postal codes to a generic phrase.

A Perl Regular Expression Macro in SAS

```
%macro PrivateUSAPostalCode(TextVar);
   &TextVar = prxchange(
     's/\d{5}/_PRIVATE_USA_POSTAL_CODE_/',
     -1,&TextVar);
%mend PrivateUSAPostalCode;
```

This is an example of how a Perl regular expression can be used for processing text. In this case, the code is eliminating identifying ZIP code information. This macro uses the SAS PRXCHANGE function to convert the occurrence of a five-digit ZIP code into the string _PRIVATE_USA_POSTAL_CODE_. You might need to perform many steps to get the text data in the form that you need.

7

Running SAS Programs

In SAS Enterprise Miner

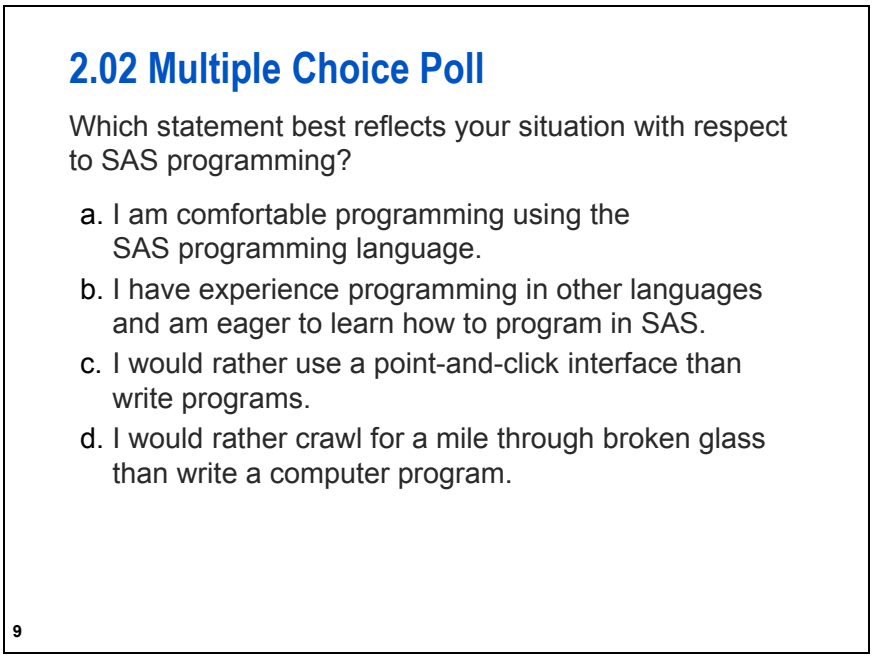
- Program Editor: **View** ⇨ **Program Editor**
- SAS Code node: Utility tab

Outside SAS Enterprise Miner

- SAS Enterprise Guide
- SAS windowing environment
- batch
- other

8

SAS programs are used sparingly in this course. Preference is given to the use of the SAS Code node for running SAS programs.



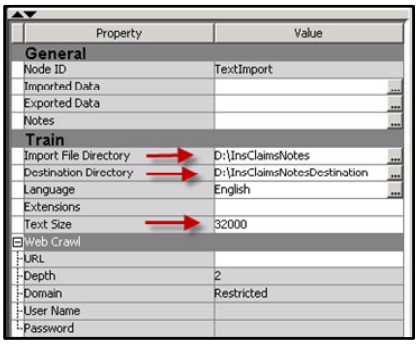
Because data preparation can be the most arduous task in text mining, SAS Text Miner includes the Text Import node that facilitates reading all popular commercial document formats.

Previous versions of SAS Text Miner included the %TMFILTER SAS macro for reading document collections. The %TMFILTER macro is available in the current release of SAS Text Miner. The Text Import node provides the functionality of the %TMFILTER macro without requiring the user to create and execute a SAS program.

A *SAS macro* is a script that can be compiled and executed to perform complex tasks. At the simplest level, a SAS macro is a script that generates SAS code to be executed by the SAS supervisor. These scripts are often stored in SAS catalogs that can be accessed and viewed by users. Proprietary scripts are stored in compiled form and cannot be read by users. Some of the SAS macros included with SAS Enterprise Miner and SAS Text Miner are compiled. Sample SAS macros are stored in the catalog **SASHELP.EMUTIL**. For example, the SAS source file **SASHELP.EMUTIL.EXTDEMO.SOURCE** provides examples of SAS Enterprise Miner functionality that can be exploited using a SAS Code node. The catalog **SASHELP.EMTEXT** might contain macros related to text mining.

The Text Import Node

The Text Import node converts collections of documents into a single SAS table. Each document then becomes a row in the SAS data set. You need to specify an import file directory and also a destination directory. You can also alter the Text Size parameter (default=100).



10

continued...

The Text Import Node

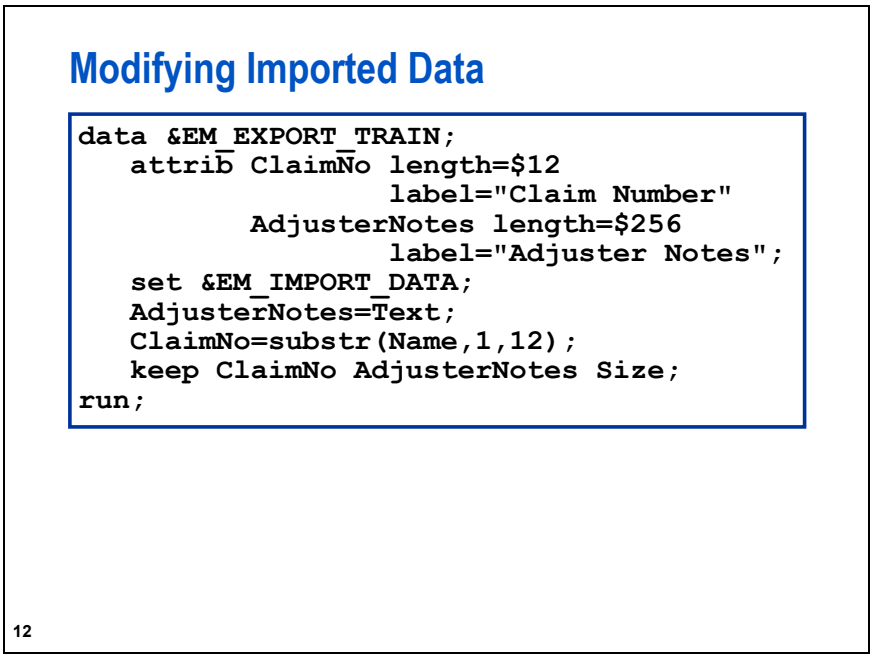
Some of the supported document types:

- Microsoft Word (.doc, .docx)
- Microsoft Excel (.xls, .xlsx)
- Microsoft PowerPoint (.ppt, .pptx)
- Rich Text (.rtf)
- Adobe Acrobat (.pdf)
- ASCII Text (.txt)
- Some others:
  - Corel
  - FrameMaker

More than 100 file formats are supported!

11

You can use a SAS Code node to modify the SAS table produced by the Text Import node. For example, you can choose to drop variables such as **LANGUAGE**, **TRUNCATED**, **OMITTED**, and **EXTENSION**, because these variables are rarely used beyond the data preparation stage. Many document collections use a naming convention so that the path given in the **URI** field or the filename given in the **NAME** field can be used to derive ID or index variables.



In the above example, the SAS Code node modifies the data produced by the Text Import node so that it can be merged with claims data indexed by the variable **ClaimNo**. This code is used in the demonstration below.

Using the Text Import Node

This demonstration starts with using the Text Import node to read in insurance adjuster notes for an insurance subrogation modeling example. The Text Import node is set up and run differently from the File Import node used in Chapter 1. You match the adjuster notes to another data set containing a target variable associated with the notes. You use a SAS Code node and follow that up with a typical text mining flow for predictive modeling. This shows many of the steps that you might follow when you work with your own data.

Some background and definitions are helpful here. The term *subrogation* refers to a legal right that an insurance company has to sue a third party in order to recover any compensation payouts. For example, if you have a car accident that is caused by some other person who was at fault for hitting you, your insurance company compensates you directly. However, it can also try to recover money from the insurance company of the person who hit you. This is called *subrogation*. Typically, it costs time and money for your company to use a lawyer to initiate subrogation proceedings, so it does not pursue this unless the company thinks that there was a good chance of winning the lawsuit. In the demonstration that follows, you work with a data set in which the target variable (**SubroFlag**) is defined by whether insurance claims were successfully subrogated.

The adjuster notes source files are contained in the **D:\workshop\winsas\dmtxt13_1\InsClaimsNotes** directory. (It is possible that the data resides elsewhere, but your instructor knows the correct pathname.)

1. Create a new diagram and name it **Subrogation Model**.

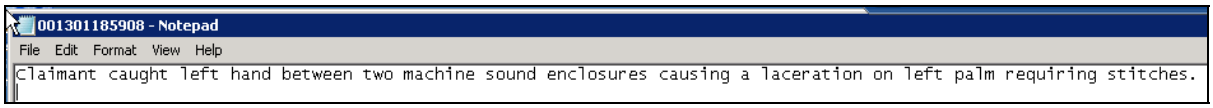
2. Drag a **Text Import** node into the diagram. For the Import File Directory property, navigate to the pathname for the insurance adjuster notes (**D:\workshop\winsas\dmtxt13_1\InsClaimsNotes**). In general, you also need to create a destination directory for the output of the Text Import node. In this course, it is already created as the **D:\workshop\winsas\dmtxt13_1\InsClaimsNotesDestination** folder. (If it does not exist, create it in Windows.) Navigate to this destination directory so that it is selected in the Properties panel. Modify the text size from the default 100 to the maximum **32000**. An example of the completed Properties panel appears below.

Property	Value
General	
Node ID	TextImport
Imported Data	
Exported Data	
Notes	
Train	
Import File Directory	D:\workshop\winsas\DMTXT13_1\InsClaimsNotes
Destination Directory	D:\workshop\winsas\DMTXT13_1\InsClaimsNotesDestination
Language	English
Extensions	
Text Size	32000

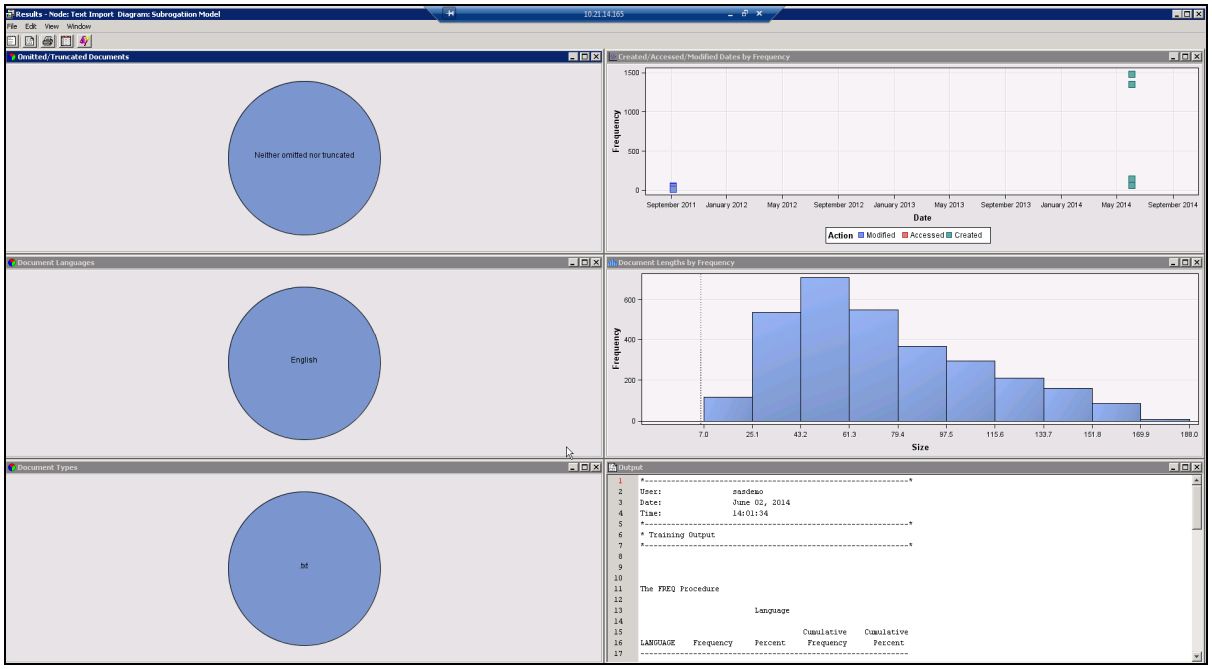
3. Look at the way that the source document files are stored in **D:\workshop\winsas\dmtxt13_1\InsClaimsNotes**. There is an individual file for each adjustor note. The names of the files are actually claims numbers and are used for matching purposes in a later step. The notes are saved as simple text documents, but the Text Import node can easily process them if they are stored as PDF, Word, and other types of files.

Name	Date modified	Type	Size
000004487308	9/5/2011 10:38 PM	Text Document	1 KB
000309831108	9/5/2011 10:38 PM	Text Document	1 KB
001301185908	9/5/2011 10:38 PM	Text Document	1 KB
001716965808	9/5/2011 10:38 PM	Text Document	1 KB
001924817308	9/5/2011 10:38 PM	Text Document	1 KB
002500385808	9/5/2011 10:38 PM	Text Document	1 KB
002525865808	9/5/2011 10:38 PM	Text Document	1 KB
002601381908	9/5/2011 10:38 PM	Text Document	1 KB
002613478908	9/5/2011 10:38 PM	Text Document	1 KB
002614936508	9/5/2011 10:38 PM	Text Document	1 KB
002701592908	9/5/2011 10:38 PM	Text Document	1 KB
002714742208	9/5/2011 10:38 PM	Text Document	1 KB
002829016508	9/5/2011 10:38 PM	Text Document	1 KB
004026028708	9/5/2011 10:38 PM	Text Document	1 KB
004726761108	9/5/2011 10:38 PM	Text Document	1 KB

4. Open one of these documents to see what it contains. For example, the file 001301185908.txt contains the sentence “Claimant caught left hand …,” which is shown below.



5. Run the **Text Import** node. View the results.



There were no omitted or truncated files. The Output window indicates that 3,037 documents were processed.

LANGUAGE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
English	3037	100.00	3037	100.00

Table of TRUNCATED by OMITTED

TRUNCATED / OMITTED	0	Total
0	3037 100.00 100.00 100.00	3037 100.00
Total	3037 100.00	3037 100.00

For example, document files are omitted if they are not one of the supported types. Truncation occurs if the document size exceeds the text size, which is 32000 bytes in this example. ***However, it is important to be aware that truncation affects only how much of the document is visible in certain windows. The full document is still analyzed downstream by all the Enterprise Miner (including Text Miner) nodes.***

6. Click the ellipsis in the **Exported Data** line in the Property panel for the Text Import node. Select **Explore** and look at the Sample Statistics window. All the variables that were created by the Text Import node are shown.

Sample Statistics

Obs #	Variable Name	Label
1	URI	URI
2	EXTENSION	Extension
3	FILTERED	Filtered
4	LANGUAGE	Language
5	NAME	Name
6	TEXT	Text
7	ACCESSED	Accessed
8	CREATED	Created
9	FILTEREDSIZE	Filtered Size
10	MODIFIED	Modified
11	OMITTED	Omitted
12	SIZE	Size
13	TRUNCATED	Truncated

After you select **Explore**, a portion of the bottom window shows three of these variables:

Text – the adjustor notes themselves

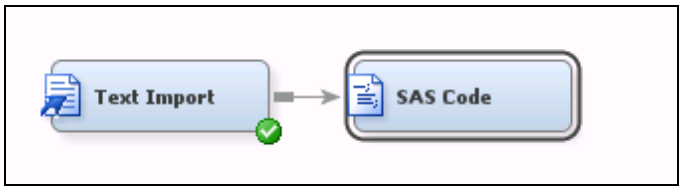
URI – the complete path to each adjustor notes file

Name – the name of the input file containing the particular adjustor note for that observation

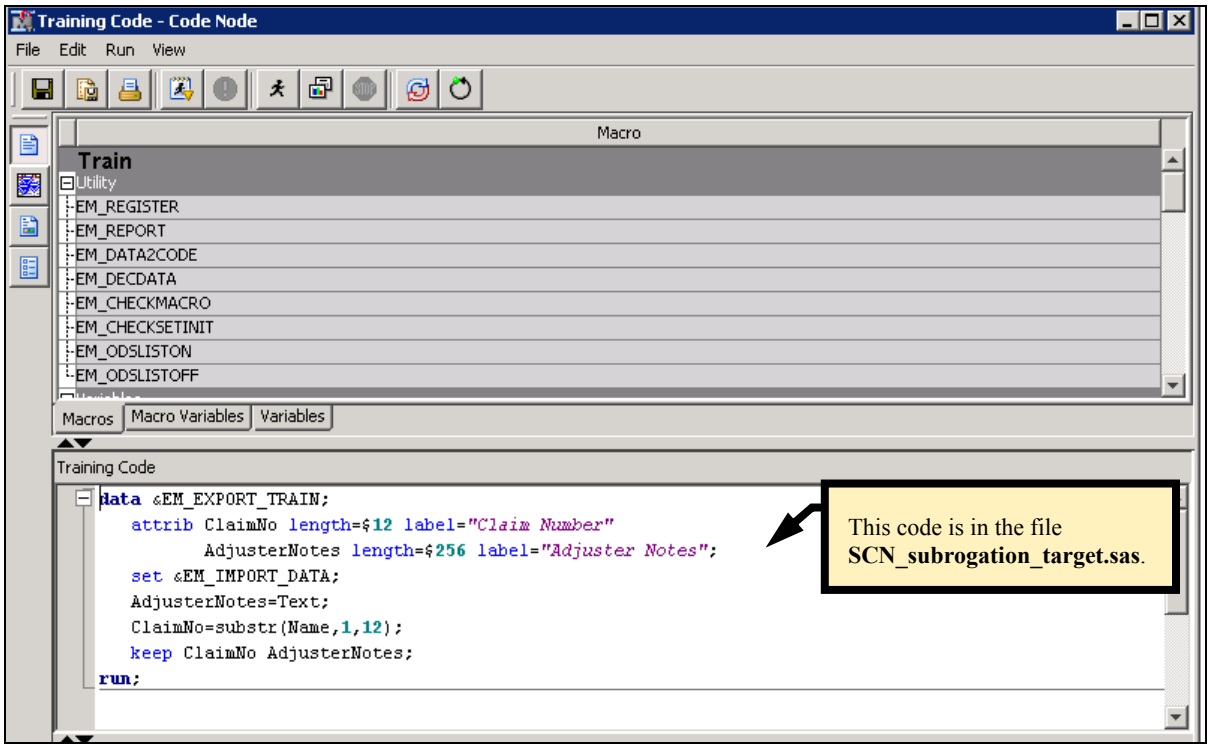
EMWS5.TextImport_TRAIN

Obs #	Text	URI	Name
1	Strained neck trying to catch falling product.	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\000004487308.txt	000004487308.txt
2	Fingers caught in machine.	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\000309831108.txt	000309831108.txt
3	Claimant caught left hand between two mac...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\001301185908.txt	001301185908.txt
4	Claimant states that while he and coworker ...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\001716965808.txt	001716965808.txt
5	Smashed right second finger, was using a d...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\001924817308.txt	001924817308.txt
6	Claimant alleges that he injured his right kn...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\002500385808.txt	002500385808.txt
7	Left ankle pain due to getting in and out of a ...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\002525865808.txt	002525865808.txt
8	While trying to avoid hitting a car out of contr...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\002601381908.txt	002601381908.txt
9	Fell in blast freezer, injured back and side.	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\002613478908.txt	002613478908.txt
10	Employee was struck by automobile --- cont...	file://D:\workshop\winsas\DMTXT13_1\InsClaimsNotes\002614936508.txt	002614936508.txt

7. Use **Name** for matching to another data set that contains the target variable **SubroFlag**. (The other data set is called **Subrogation_target**.) The name for each file is purposely set up to be the claim number. Bring in a **SAS Code** node and attach it to the **Text Import** node.

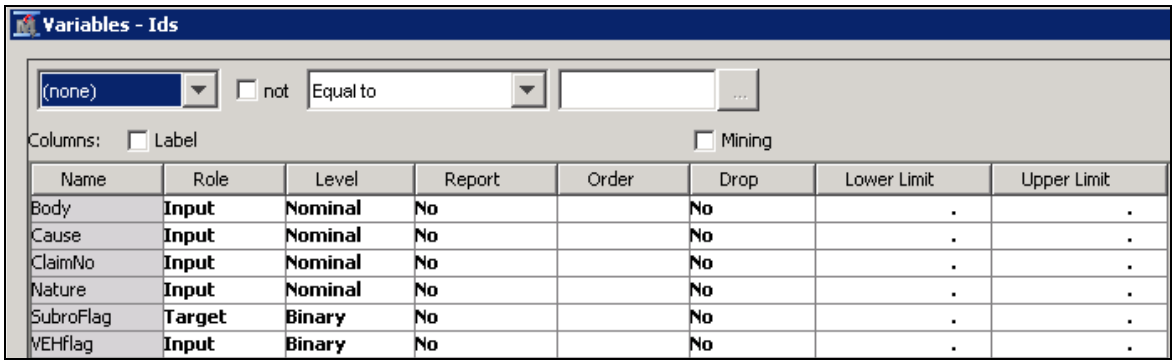


8. Select the Code Editor ellipsis in the Property Sheet for the SAS Code node. In the Training Code window, right-click and select **Open**. Then navigate to the SAS program **D:\workshop\winsas\dmtxt13_1\sassrc\SCN_subrogation_target.sas** and bring it in.



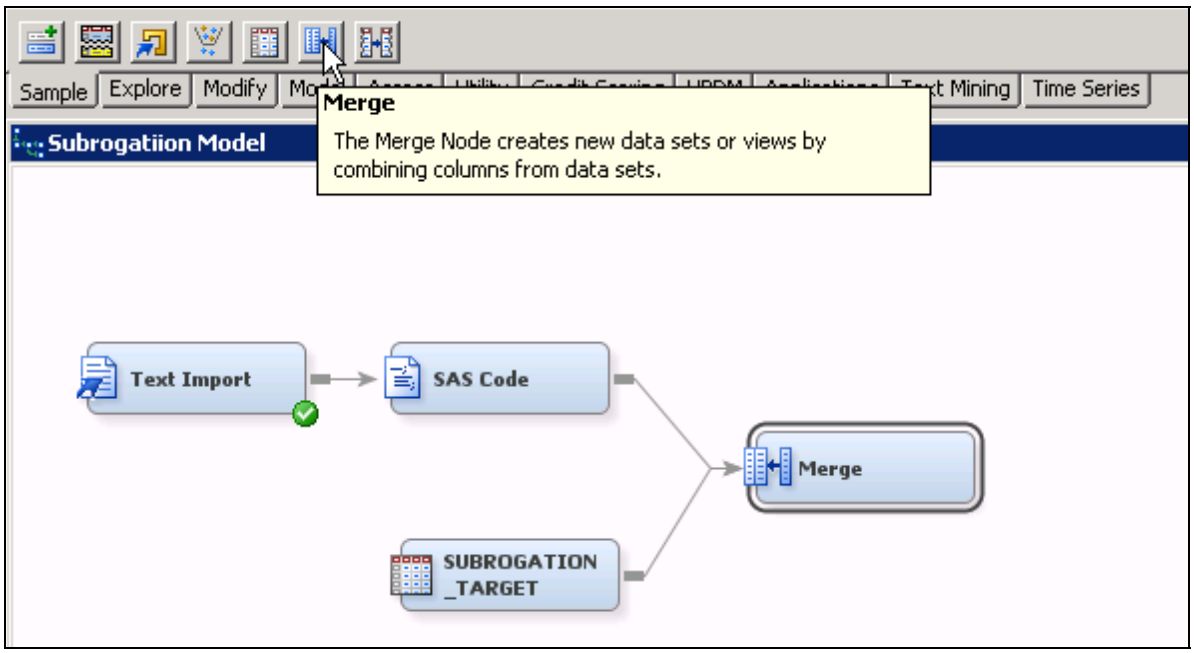
(If you have any problem finding the program, you can enter the seven lines of code manually.) This code uses **&EM_IMPORT_DATA**, a macro variable that refers to the SAS data set created by the Text Import node. The variable **AdjusterNotes** is defined in the **Text** field. (This is not really necessary because you can keep the name **Text**.) The tricky part is to define the variable **ClaimNo**, which is obtained using the SUBSTR function by extracting the first 12 characters from the variable **Name**. This is why it is essential to give each text file a name that is the same as the claim number. Run the SAS Code node.

9. Bring the SAS data set **SUBROGATION_TARGET** into the diagram. The data set was created and is shown in the Project panel under Data Sources. Look at the variables for this data set. **SubroFlag** is the binary target variable (1=successful subrogation, 0=unsuccessful). **ClaimNo** is the variable to use for matching against the adjuster notes data that were created with the Text Import node. The remaining variables are potential input variables that can be used to predict the target and the variables that the text mining nodes create.



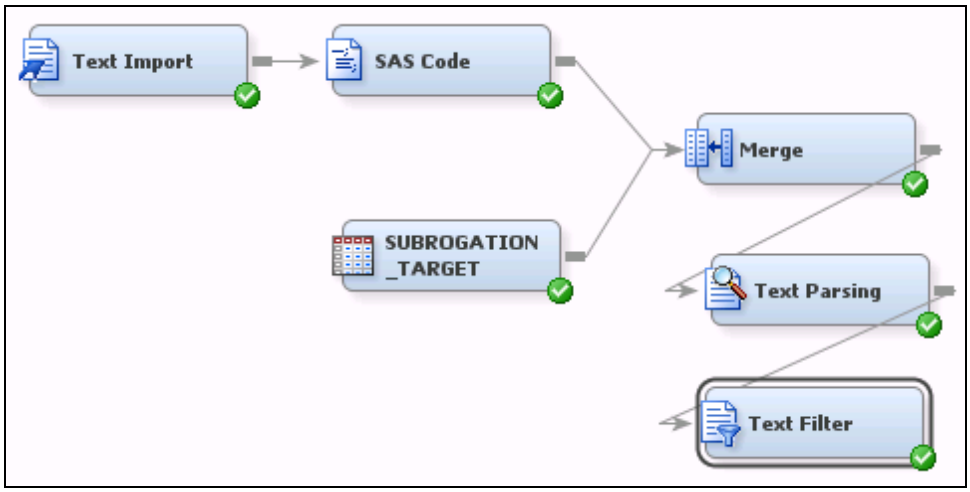
Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
Body	Input	Nominal	No		No	.	.
Cause	Input	Nominal	No		No	.	.
ClaimNo	Input	Nominal	No		No	.	.
Nature	Input	Nominal	No		No	.	.
SubroFlag	Target	Binary	No		No	.	.
VEHflag	Input	Binary	No		No	.	.

10. Bring in the **Merge** node and connect it to the two data sets. They are matched by **ClaimNo**. This node is found on the Sample tab at the top of the window.



The data set **SUBROGATION_TARGET** is already ordered by **ClaimNo**. It is matched one-to-one with the SAS data set created by the **Text Import** node and the program in the SAS Code node. Run the program from the Merge node.

11. Connect a **Text Parsing** node and a **Text Filter** node to the output of the **Merge** node. Run the default settings in both cases. For the Text Filter node, the default weightings on the Property panel are **Log** for **Frequency Weighting** and **Mutual Information** for **Term Weight**, if there is a target variable present. (There is in this case.) It is usually a good idea to explicitly show these choices rather than keep **Default** showing. Run the program from the Text Filter node.

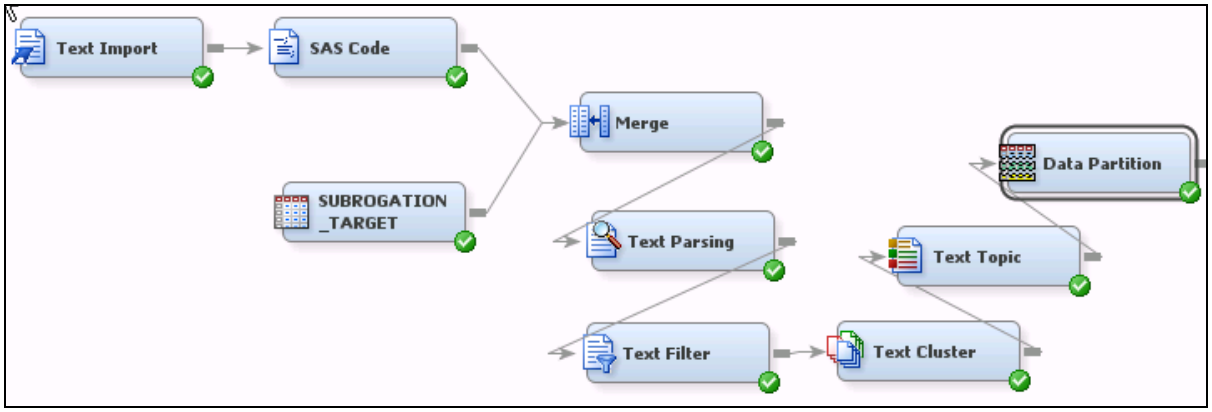


Weightings	
Frequency Weighting	Log
Term Weight	Mutual Information

Text Filter Settings

12. Connect a **Text Cluster** node to the **Text Filter** node and a **Text Topic** node to the **Text Cluster** node. Use the default settings for both.

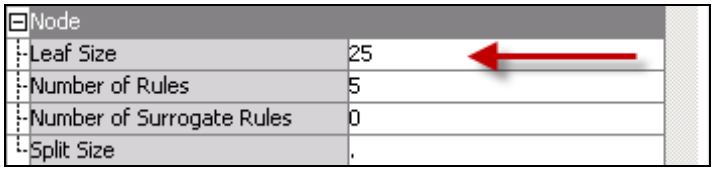
13. Connect a **Data Partition** node to the output of the **Text Topic** node. Keep the default partition percentages of 40%, 30%, and 30% for Training, Validation, and Test respectively. The diagram should appear as follows:



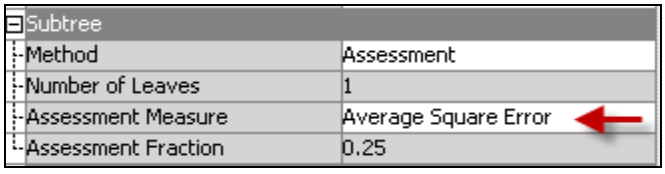
14. Use two separate decision trees to produce two prediction models for the target variable, **SubroFlag**. One decision tree uses ***all*** the available input variables, including the text mining variables created by the Text Cluster and Text Topic nodes. The second decision tree does ***not*** use any of the text mining variables. By comparing these two models, you can determine the incremental model improvement using the text mining variables.

a. Connect a **Decision Tree** node to the **Data Partition** node. The Decision Tree node automatically considers all available input variables, including the text mining variables. As discussed in the demonstration of Chapter 1, the text mining variables by default consist of the **TextCluster_SVD** variables and the **TextTopic_Raw** variables. Change two default parameters on the Decision Tree node.

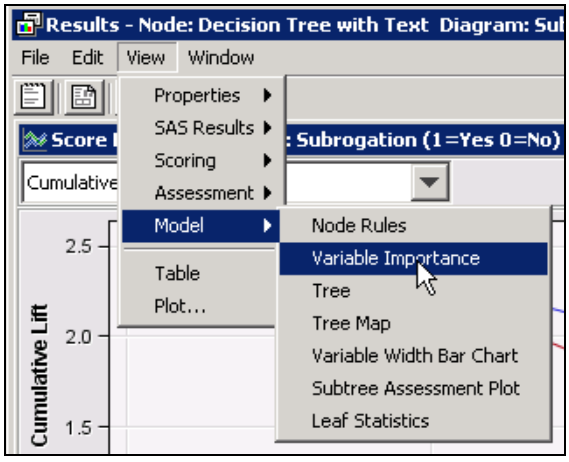
1) The leaf size is increased from the default of 5 to **25**. This parameter specifies the minimum leaf size that can be created with the tree. Many analysts prefer to have a larger number because it helps ensure that the tree model generalizes well to new data.



2) Change the **Assessment Measure** from **Default** to **Average Square Error**. (Typically, **Average Square Error** works very well as a general metric for pruning decision trees on the Validation data.)



15. Rename the first Decision Tree node **Decision Tree with Text**. Run it and then open the Results window. Select **View** ⇨ **Model** ⇨ **Variable Importance** as shown below.



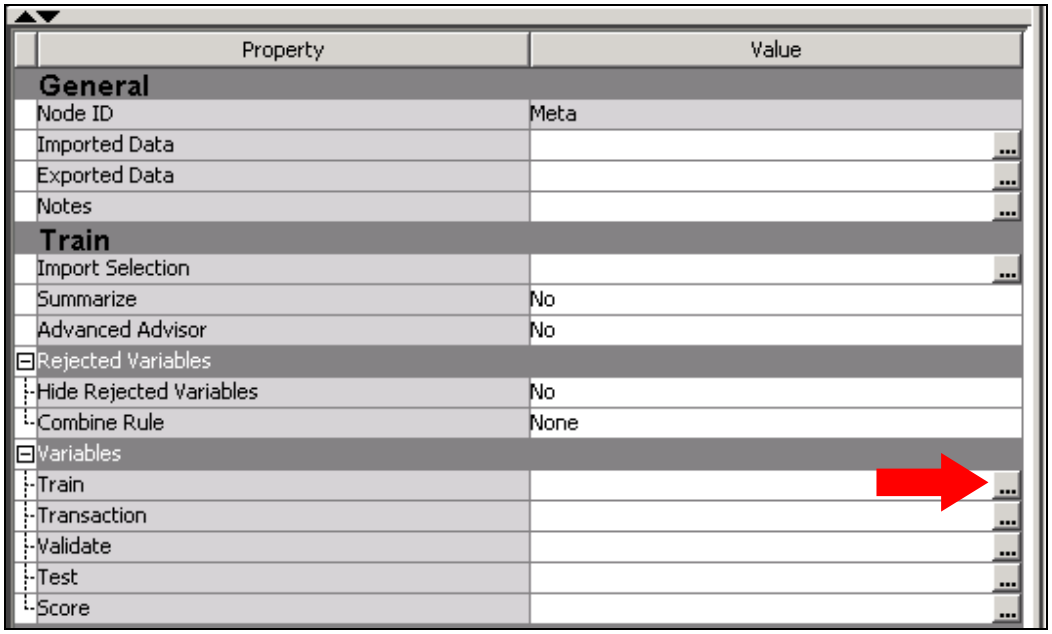
The Variable Importance window shows the variables that are used to create the tree model. The most important variable is **Cause** (**Cause of Injury**), which is not one of the text mining variables. However, all the remaining variables selected by the tree algorithm (**TextCluster_SVD13**, **TextTopic_raw12**,…, **TextCluster_SVD9**) are text mining variables.

Variable Importance

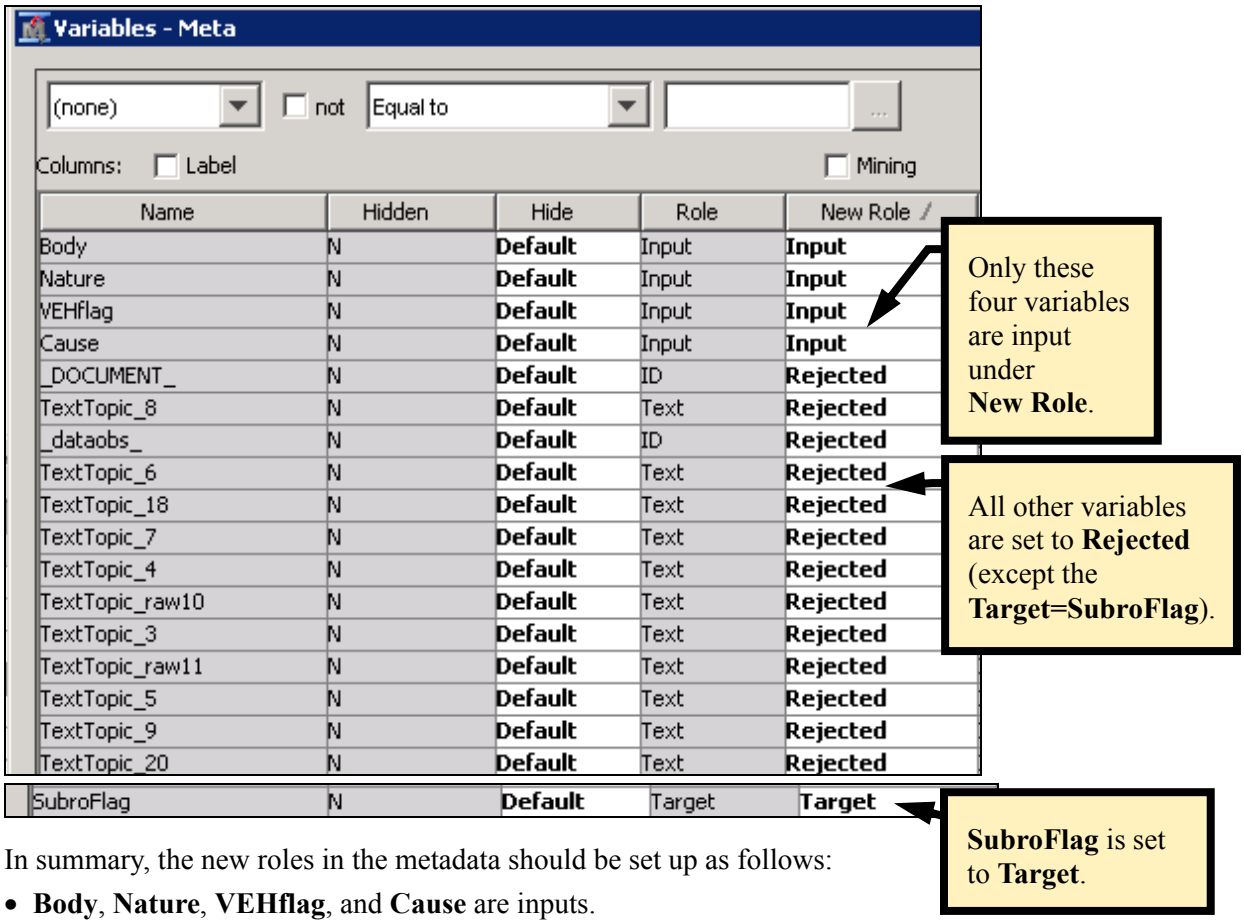
Variable Name	Label	Number of Splitting Rules	Importance
Cause	Cause of Injury	1	1.0000
TextCluster_SVD13		1	0.5772
TextTopic_raw12	+cause,+laceration,+bruise,+employee,carpal	1	0.3859
TextCluster_SVD5		1	0.3559
TextCluster_SVD1		2	0.3531
TextCluster_SVD14		1	0.2887
TextTopic_raw20	+contusion,+state,fell,wet,+floor	1	0.2696
TextCluster_SVD11		1	0.2379
TextCluster_SVD19		1	0.2280
TextCluster_SVD9		1	0.1974

Close the results for this tree.

16. Run a second decision tree. Do not use any of the text mining variables. To set this up, open a Metadata node and connect it to the output of the **Data Partition** node. Go to the **Train** property on the Property Sheet for the metadata.



Inside the Train window, set the new role for these four variables to **Input**: **Body**, **Cause**, **Nature**, and **VehFlag**. Make sure that the new role of **SubroFlag** is **Target**. All the other variables should have a new role of **Rejected**.

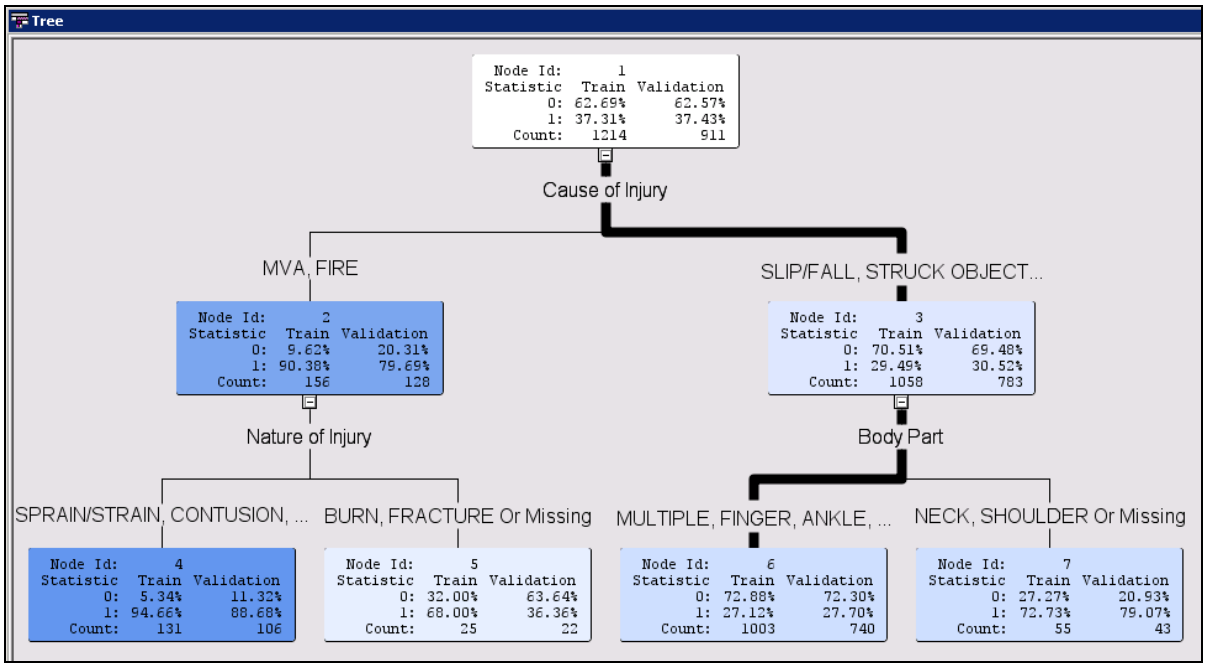


In summary, the new roles in the metadata should be set up as follows:

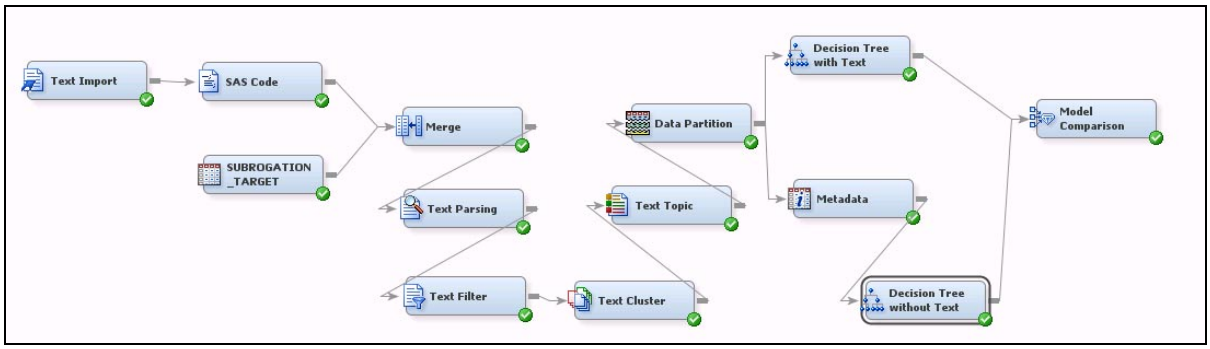
- **Body**, **Nature**, **VEHflag**, and **Cause** are inputs.
- **SubroFlag** is the target.
- All other variables have **Rejected** as the new role.

17. Open a second Decision Tree node, and connect it to the **Metadata** node. Rename it **Decision Tree without Text**. Set up the parameters the same way as for the other decision tree (that is, **Leaf Size=25** and **Assessment Measure=Average Squared Error**).

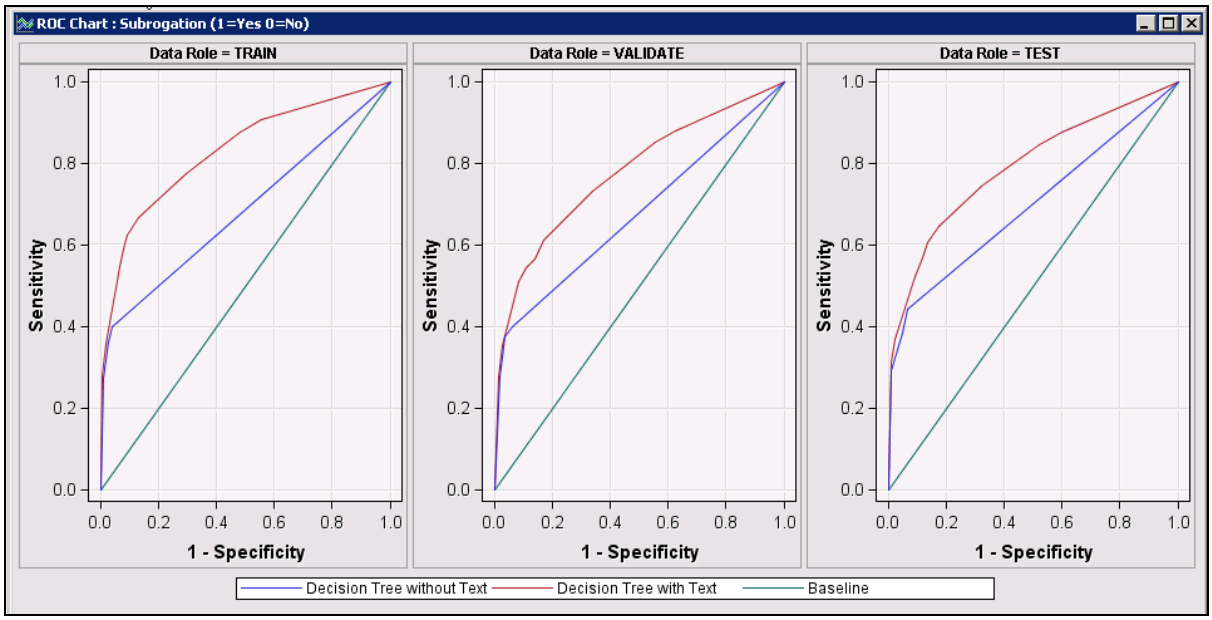
    Run the **Decision Tree without Text** node and confirm that no text mining variables were used. The final tree has four leaves and used the three variables (**Cause of Injury**, **Nature of Injury**, and **Body Part**).



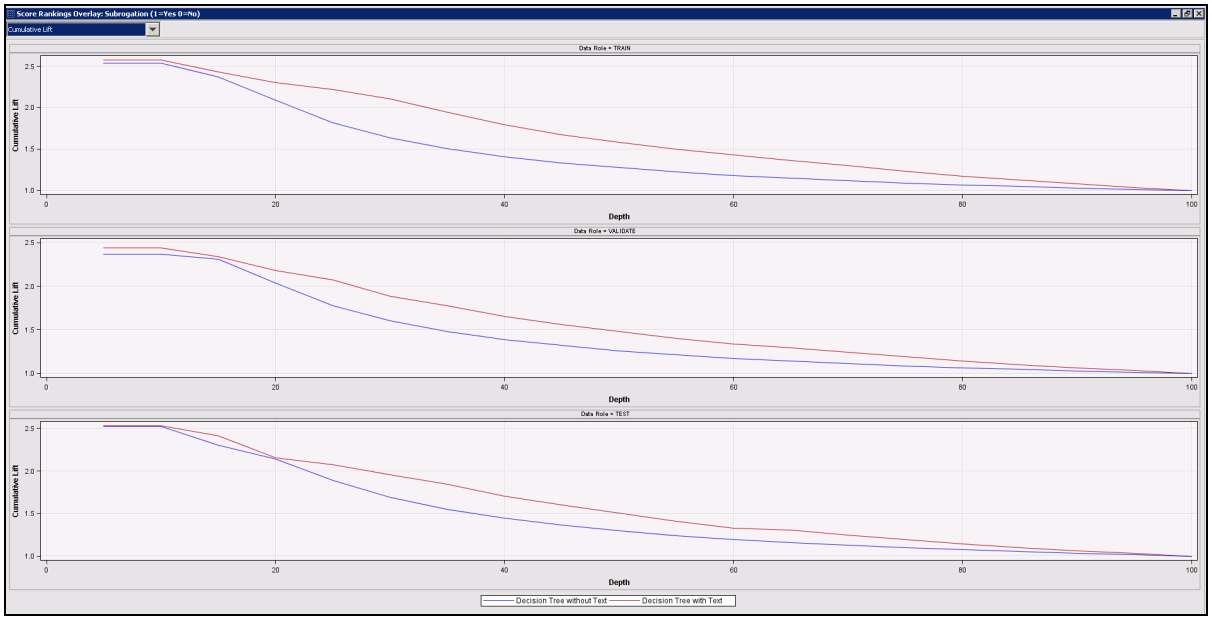
18. The last step compares the two tree models to see how much the text mining variables improved various metrics of model performance. Open a **Model Comparison** node and connect the output of both trees to it. The final setup for this entire flow should be as follows:



Run the Model Comparison node and look at the Results window. The ROC graphs in the upper left corner are consistent across the **Training**, **Validation**, and **Text** data sets. All three show that the model that included the text mining variables exhibited substantially better performance.



Similarly, the three cumulative lift charts in the lower left corner of the Results window also show substantially better performance for the model with text mining variables.



Finally, looking at the Output window for the **Test** data set, the first three key metrics indicate better results for the first tree compared to the second tree (higher Kolmogorov-Smirnov and ROC index, lower Average Squared Error).

Data Role=Test

Statistics	Tree	Tree2
Test: Kolmogorov-Smirnov Statistic	0.47	0.38
Test: Average Squared Error	0.17	0.18
Test: Roc Index	0.79	0.70

This demonstration illustrated how to create a corpus of documents using the Text Import node and how to do the additional data processing to incorporate additional input and target variable information. The subsequent text mining nodes (Test Parsing, Text Filter, Text Cluster, and Text Topic) were then run using default settings. Two decision trees (one with text mining variables and one without) were run and compared. This entire diagram flow probably resembles, in a general way, many of the steps that you are likely to use in your work environment.

Exercises

1. Interpreting the Document Clusters in Relation to Other Variables

Interpreting results from text mining can be challenging. In particular, the SVD dimensions are rarely transparent. However, the clusters, which are derived from the SVD dimensions, can be very helpful in understanding your results. Recall that the Text Cluster node generates the cluster variable **TextCluster_cluster_**. This variable is produced by using the SVD variables derived from the Adjustor Nodes to cluster the documents. With the subrogation data, running the default Text Cluster node gives eight clusters with the following descriptive terms:

Clusters

Cluster ID	Descriptive Terms	Frequency
1	+fall +floor +lift +low +stair back +'low back' wet +slip +walk ladder walking felt +strain +box ...	393
2	'auto accident' 'motor vehicle accident' +involve accident auto motor vehicle 'vehicle accident' 'car accident' neck +injury...	209
3	'left shoulder' +car +injury +shoulder company driving neck vehicle +back rearended +drive +strain accident side +st...	463
4	+cause +eye +wrist +hand metal +work machine +twist left +object 'left hand' +step right +claimant +leg ...	750
5	+strike +tree head front +object fracture +door +rack 'left arm' forklift face +contusion cart +laceration side ...	69
6	'left index finger' +'index finger' +catch +cut +door +finger +laceration +machine +thumb index knife smashed 'left thu...	384
7	+employee +state walking +twist +pallet +foot +allege +ankle +move +slip ladder strained +truck side +knee ...	352
8	+contusion +injure +leg +worker fell +bruise +knee +back +claimant back +walk forklift +low side +floor ...	417

The descriptive terms provide interpretable information about which documents are likely to occur within the clusters. With the subrogation data, you also have a number of other variables, such as **Body** (location of the injury), **Cause** (cause of the injury), **Nature** (nature of the injury), and **VEHFlag** (whether a vehicle was involved). They can be used to provide more interpretation. These variables were ***not*** used to produce the clusters (only the SVD dimensions were), but they can be used to clarify the clustering solution.

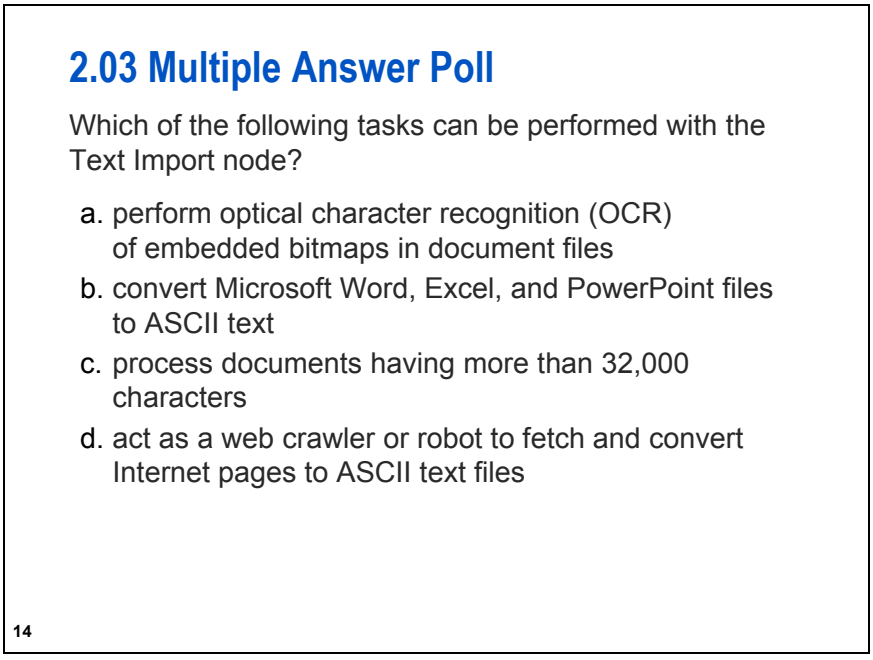
a. Attach a **SAS Code** node to the output of the **Text Cluster** node. Set up **the SAS Code** node so that it does a crosstabulation between the **TextCluster_cluster_** variable the non-text categorical variables **Body**, and the target variable, **SubroFlag**. Looking at the crosstabulation with **SubroFlag**, which two clusters have the ***strongest*** association with a successful subrogation (that is, **SubroFlag**=1)? Can you see from the descriptive terms of these two clusters what is likely to be a common theme in the adjuster notes?

b. Which cluster of documents had the ***weakest*** association with a successful subrogation? Based on the descriptive terms for this weakest cluster, what is a primary characteristic of these documents? Is the crosstabulation of **TextCluster_cluster_ * Body** (the body part involved with the accident) consistent with this interpretation?

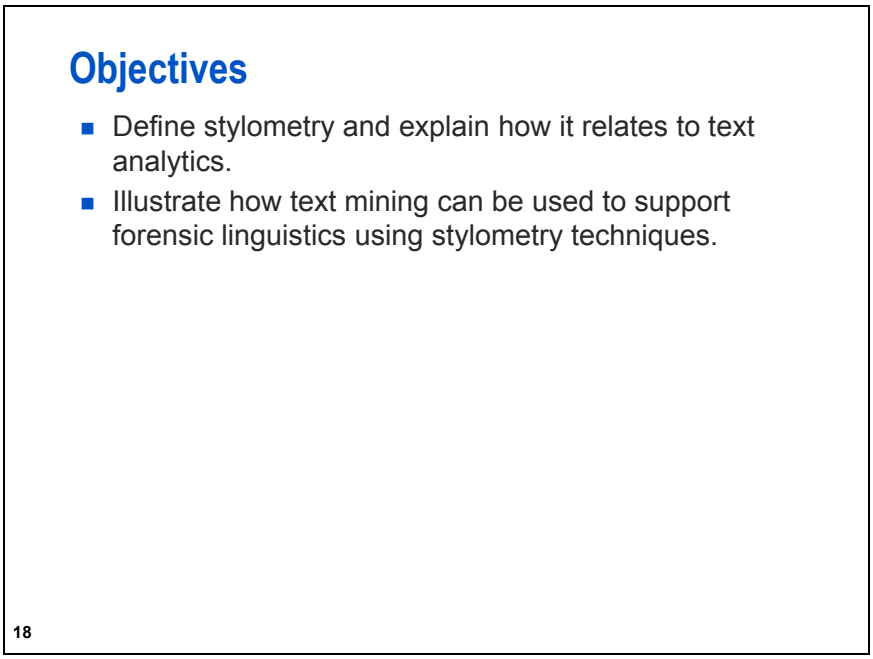
c. Read the following Adjustor Note. Do you think this claim would likely be successful in subrogation?

"Claimant trying to pry pallet jack from in between dock and trailer where it was stuck, slipped and smashed finger in between the two."

The purpose of this exercise is to indicate how to interpret the **Text_cluster_** variable when other non-text variables are available to help you do this. In this case, only the variables **SubroFlag** and **Body** were used for this purpose. However, other non-text variables are available (**VEHflag**, **Nature**, **Cause**), which can also be used to better understand the **Text_Cluster_** variable. In addition, the **Profile Segment** is useful for further exploration of clusters. Consider using that to help you better understand the clusters that were created.



2.2 A Forensic Linguistics Application



SAS users come from many different business and government organizations. Students in this class are sometimes involved with various security and intelligence problems. The forensic linguistics demonstration is intended to show how SAS Text Miner can be used for these types of problems. Consider some background information for this example. Between 1978 and 1995, a person called the "Unabomber" (who is known to be Theodore Kaczynski) mailed bombs to selected individuals associated with technology research. His bombs killed three people and injured 23. In 1995, he sent a long article entitled *Industrial Society and Its Future* to the FBI and demanded that it be published in a major newspaper or he would strike again. This long article was eventually published in the *New York Times* and *The Washington Post*. The style and content of the writing was recognized by Theodore Kaczynski's brother, and this ultimately led to Kaczynski's arrest.

This demonstration uses 232 paragraphs extracted from Kaczynski's long article, and 1726 paragraphs extracted from the writing of five other authors. The latter are used as "comparison" documents. There is a total of 1958 documents (paragraphs). You run both the Text Cluster and Text Topic nodes on this training data and then create a decision tree model in order to attempt to accurately classify the documents by their authors. Classification such as this is really a form of prediction modeling. In addition, 11 documents are used as unknowns. You use the two models to classify these 11 unknown paragraphs with regard to their likely authors. (Spoiler alert: In this setup, all 11 of the unknown cases were written by Kaczynski.)

Stylometry

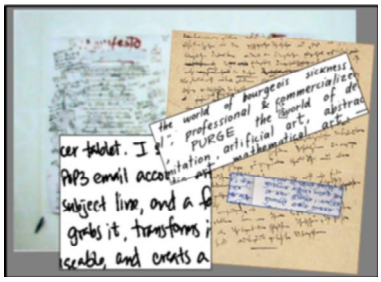
Stylometry is defined as the use of linguistic style to characterize written language.

Applications:

- attributing authorship of anonymous or disputed literary works
- detecting plagiarism
- forensic linguistics, for example, identifying Theodore Kaczynski as the Unabomber based on his writing style

19

Forensic Linguistics



Special Case: Stylometry Applied to Forensics

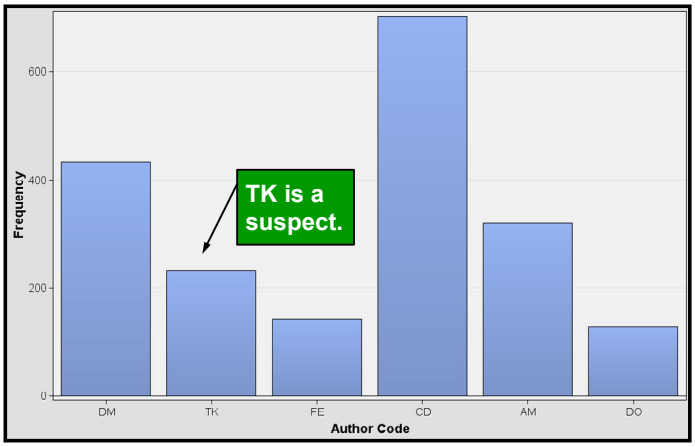
Problem: Eleven written sources… Who is the author?

20

continued...

Forensic linguistics typically uses predictive modeling to score a document of unknown, but suspected, authorship. The score represents an estimate of the probability that the document was written by a suspect. The value of text mining applied to forensic linguistics is that suspects can be identified for investigation. The text mining results are rarely if ever used as evidence in prosecuting a suspect, although testimony might include a discussion of techniques in describing how the suspect was identified.

Forensic Linguistics



Corpus: 1,958 paragraphs from six authors taken from written works and interviews

21

continued...

The data for this study is real, but the situation is hypothetical. Separation of documents was enhanced for educational purposes. In actual forensic linguistic studies, there are rarely such pure results as those achieved here.

The six authors in the training data are coded as AM, CD, DM, DO, FE, and TK. The initials were changed for the first five authors. TK is Theodore Kaczynski, the so-called Unabomber. The TK documents are paragraphs from the manifesto written by Kaczynski and published in *The New York Times* and *Washington Post*. Obviously, when the manifesto was published, the author was not known to be TK. The 11 unknown documents are excerpts from interviews with Kaczynski after he was convicted of murder. Thus, although based on real data, this example is artificial.

Forensic Linguistics

EMWS13.Ids2_DATA

Obs #	Pa...	Extracted Text
1	6001	I read Edward Abbey in mid-eighties and that was one of the things that gave me the idea that,...
2	6002	Back in the sixties there had been some critiques of technology, but as far as I knew there wer...
3	6003	The honest truth is that I am not really politically oriented. I would have really rather just be livin...
4	6004	Unquestionably there is no doubt that the reason I dropped out of the technological system is ...
5	6005	Many years ago I used to read books like, for example, Ernest Thompson Seton's "Lives of Ga...
6	6006	I have quite a bit of experience identifying wild edible plants, it's certainly one of the most fulfilli...
7	6007	One thing I found when living in the woods was that you get so that you don't worry about the fu...
8	6008	The best place, to me, was the largest remnant of this plateau that dates from the tertiary age. ...
9	6009	I don't think it can be done. In part because of the human tendency, for most people, there are ...
10	6010	The big problem is that people don't believe a revolution is possible, and it is not possible pre...
11	6011	While I was living in the woods I sort of invented some gods for myself. Not that I believed in th...

Score Data Set: Eleven documents from the same unknown author

Problem: Build classification models on the known documents with six different authors. Apply these models to the unknown 11 documents to determine the likely author of each one.

22

continued...

Stylometry for Forensic Linguistics

This demonstration illustrates how to use text mining nodes and other Enterprise Miner nodes to build classification (prediction) models in a forensic setting. You analyze writing samples from six authors. For five authors, the writing samples in the training data have to do with technical material about statistics and SAS courses. For one of the authors (TK), the writing samples are paragraphs from his published manifesto.

1. Create a diagram named **Forensic Linguistics**.

2. The training data that is used is contained in one SAS data set called **Forensics**. It resides in the directory **D:\workshop\winsas\DMTXT13_1**. For this demonstration, the document (paragraph) extracts were already run through **Text Import** and a target variable (**Author**) was associated with each document. Therefore, first create a library pointing to this directory. Select **File** ⇨ **New** ⇨ **Library**. Then navigate to **D:\workshop\winsas\DMTXT13_1**. Specify the library name as **DMTXT**.

3. Go to **Data Sources** for the project. Right-click to open **Create Data Source**. Select **Browse** to find the SAS data set **Forensics** in the **DMTXT** library. Select **Forensics**. In step 5 of the Data Source Wizard when the variables are shown, change the variable roles to correspond to the following:

Name	Role	Level
Author	**Target**	**Nominal**
DocID	**ID**	**Nominal**
Text	**Text**	**Nominal**

   Bring the data set into the diagram.

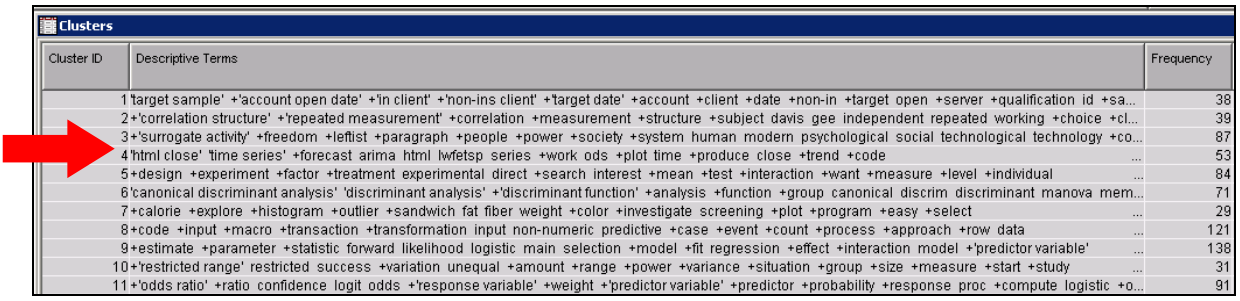
4. Connect a **Data Partition** node to the **Forensics Input Data** node and retain the default settings (40%/30%/30%).

5. Connect a default **Text Parsing** node to a **Data Partition** node.

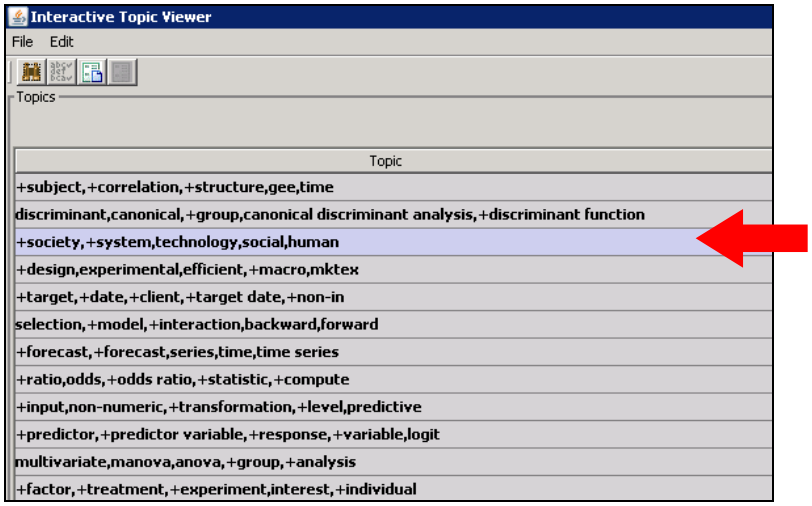
6. Connect a default **Text Filter** node. Change the **Weightings** to explicitly show **Log** and **Mutual Information**. Remember, these are actually the defaults that are used here because a nominal target variable is present. It is always a good idea to make this clear in the Property Panel.

Weightings	
Frequency Weighting	Log
Term Weight	Mutual Information

7. Attach a **Text Cluster** node to the **Text Filter** node and run it with the default settings. Open the results. Cluster 3 has descriptive terms, such as *power*, *society*, and so on, that are clearly associated with the Unabomber's long published manifesto. Close the window.

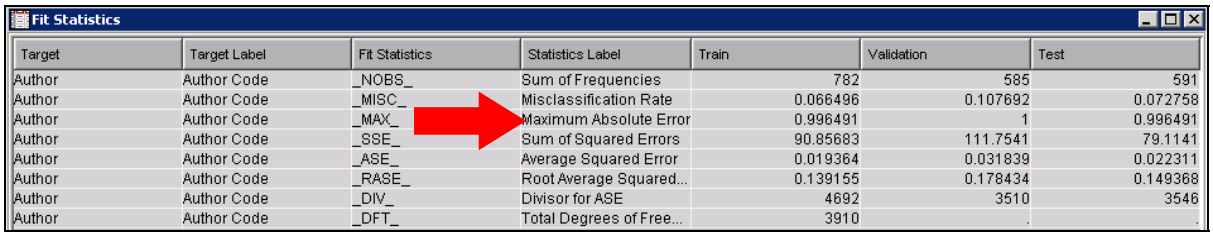


8. Attach a default **Text Topic** node to the **Text Cluster** node and run it. Open the **Interactive Topic Viewer** and look for topics that are likely to be from the Unabomber's writing. For example, select the third topic shown below. Look at the Terms and Documents windows associated with this topic.

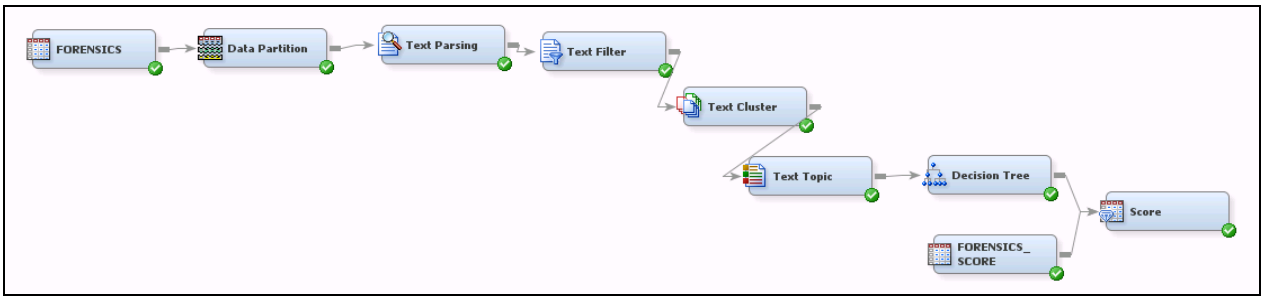


Close **the Interactive Topic Viewer**.

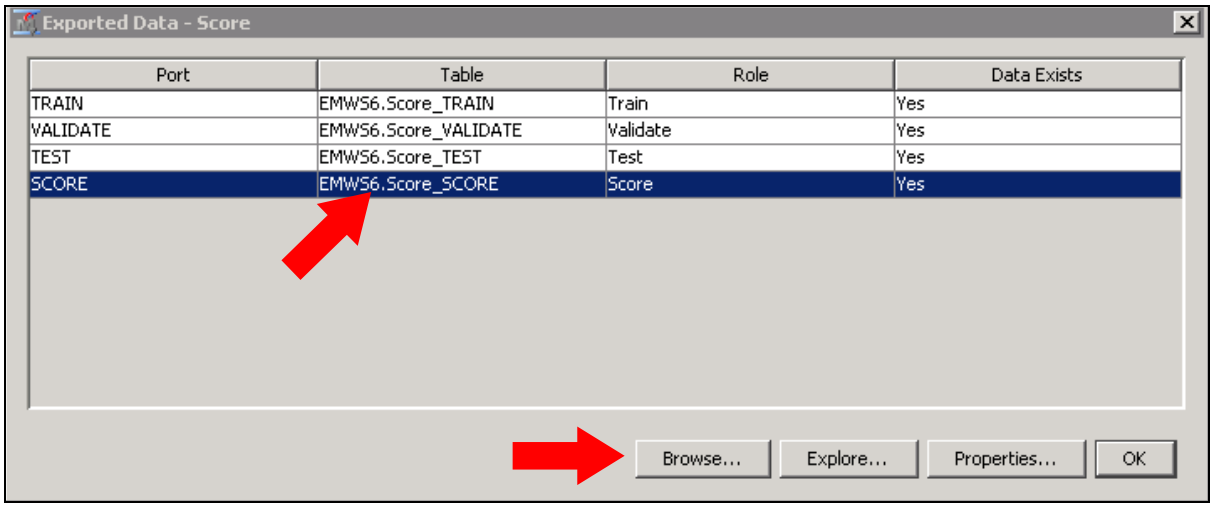
9. Connect a **Decision Tree** node to the **Text Topics** node. Change the default leaf size from 5 to **25** and change the assessment measure to **Average Square Error**. Run the tree and look at the tree in the Results window. Notice how well the leaves of the tree separate the six authors. In particular, the author TK is accurately classified. The overall misclassification rate for all the document extracts can be seen in the Fit Statistics window. These rates are approximately .066, .108, and .073 for the **Train**, **Validation**, and **Test** data sets, respectively. Clearly, good accuracy was achieved.



10. Open the **Forensics_score** data set and designate it as a **Score** data set. This data set contains the 11 paragraphs that were drawn from TK's interview after he was captured. You want to see how accurately the tree model classifies these paragraphs. To do this, open a **Score** node and attach it to both the **Forensics_score** data set and the **Decision Tree** node. Your complete diagram should look as shown below.



Run the **Score** node and open the **Exported Data** from the Property Panel. Select the **Score** data and click **Browse**.



Scroll to the far right in the browsing window and look at the last two columns.

Probability of Classification	Prediction for Author
1.0	TK
1.0	TK
1.0	TK
0.84	TK
0.84	TK
1.0	TK
1.0	TK
1.0	TK
1.0	TK
1.0	TK
0.84	TK

The last column gives the model's predicted author category (Prediction for Author) and the second to last column gives the model probability for this category (Probability of Classification). All 11 paragraph extracts are correctly classified as written by TK. (Remember, the data for this demonstration was enhanced to ensure such a clear-cut result!)

2.3 Information Retrieval

Objectives

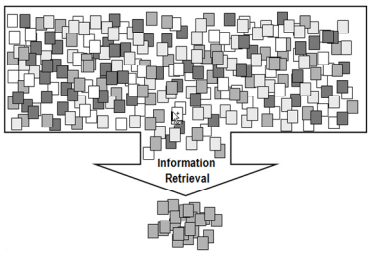
- Describe information retrieval and explain how it is done in the interactive Text Filter Viewer.
- Use the Medline medical abstracts data to illustrate an application of information retrieval.

28

Information Retrieval

"Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)."

– Manning, Raghavan, and Schütze (2008)



29

One of the more publicized success stories in information retrieval concerns the discovery by Don Swanson (1988, 1991) that magnesium deficiency could be a source of migraine headaches. Swanson queried medical reports for articles about migraines and nutrition.

For a given corpus of documents, information retrieval (IR) groups documents based on the similarity of contents. An IR query can be a Boolean query, a query based on latent semantic indexing, or a query based on some other method of quantifying document content. The Text Filter node uses a weighted *cosine similarity* measure to compute the similarity between a document and the query. Documents that are most similar to the query are returned.

Filtering and Querying

Filtering and querying using the Interactive Filter Viewer:

- Query operators control how filtering is performed.
- To clear a query, select **Clear** ⇨ **Apply**.
- You can close the Interactive Filter Viewer and save the current query. Rerunning the node exports the results of the query rather than the full data set.

30 *continued...*

Filtering and Querying

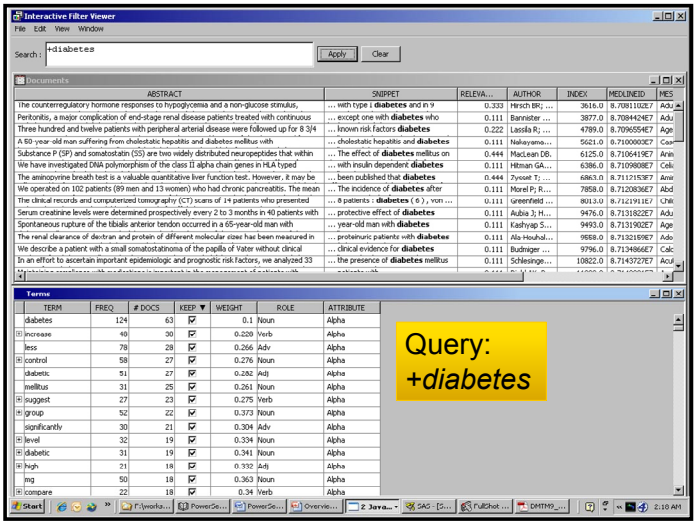
Text Filter Query Operators (Review)

- *+term* returns all documents that have at least one occurrence of *term.*
- *-term* returns all documents that have zero occurrences of *term.*
- "*text string*" returns all documents that have at least one occurrence of the quoted text string.
- *string1*string2* returns all documents that have a term that begins with *string1*, ends with *string2*, and has text in between.
- *>#term* returns all documents that have *term* or any of the synonyms that were associated with *term.*

31 *continued...*

The Interactive Filter Viewer does not recognize ># operators mixed with the + operator.

Filtering and Querying



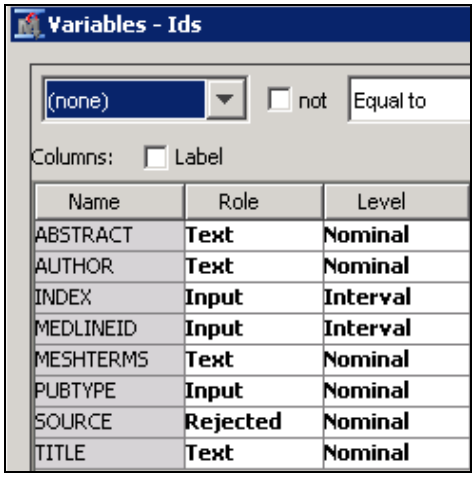
32

Retrieving Medical Information

You often want to explore a document collection by searching on various terms of interest. This does not require a target variable and is efficiently done with the Text Filter node. As always, you first run the Text Parsing node. This demonstration illustrates how to do this using medical information from Medline data.

The **MEDLINES** data source contains a sample of 4,000 abstracts from medical research papers that stored in the MEDLINE data repository.

1. Create a new diagram and name it **Medline Information Retrieval**. Drag the **MEDLINES** data source into the diagram. Look at the variables.

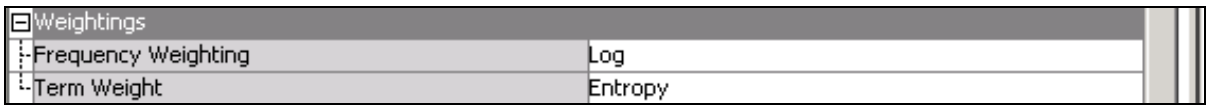


Name	Role	Level
ABSTRACT	Text	Nominal
AUTHOR	Text	Nominal
INDEX	Input	Interval
MEDLINEID	Input	Interval
MESHTERMS	Text	Nominal
PUBTYPE	Input	Nominal
SOURCE	Rejected	Nominal
TITLE	Text	Nominal

There is more than one variable with the role of **Text**. In cases like this, the Text variable with the longest length is the one that is selected for analysis by the Text Parsing node. If two or more Text variables have the same length, the one appearing first in alphabetical order is selected. In this example, **ABSTRACT** (2730 bytes in length) is the longest of the Text variables and is the one that is analyzed.

2. Attach a default **Text Parsing** node to the **Input Data Source** node. Notice that the default Text Parsing node populates the Properties Panel with certain tables. For example, there is a default Synonyms table called **SASHELP.ENGSYNMS**. (This actually contains only one row (one synonym) and is present only as a template). There is also a default Stoplist called **SASHELP.ENGSTOP**. (The use of such tables and others is discussed in a later chapter.)

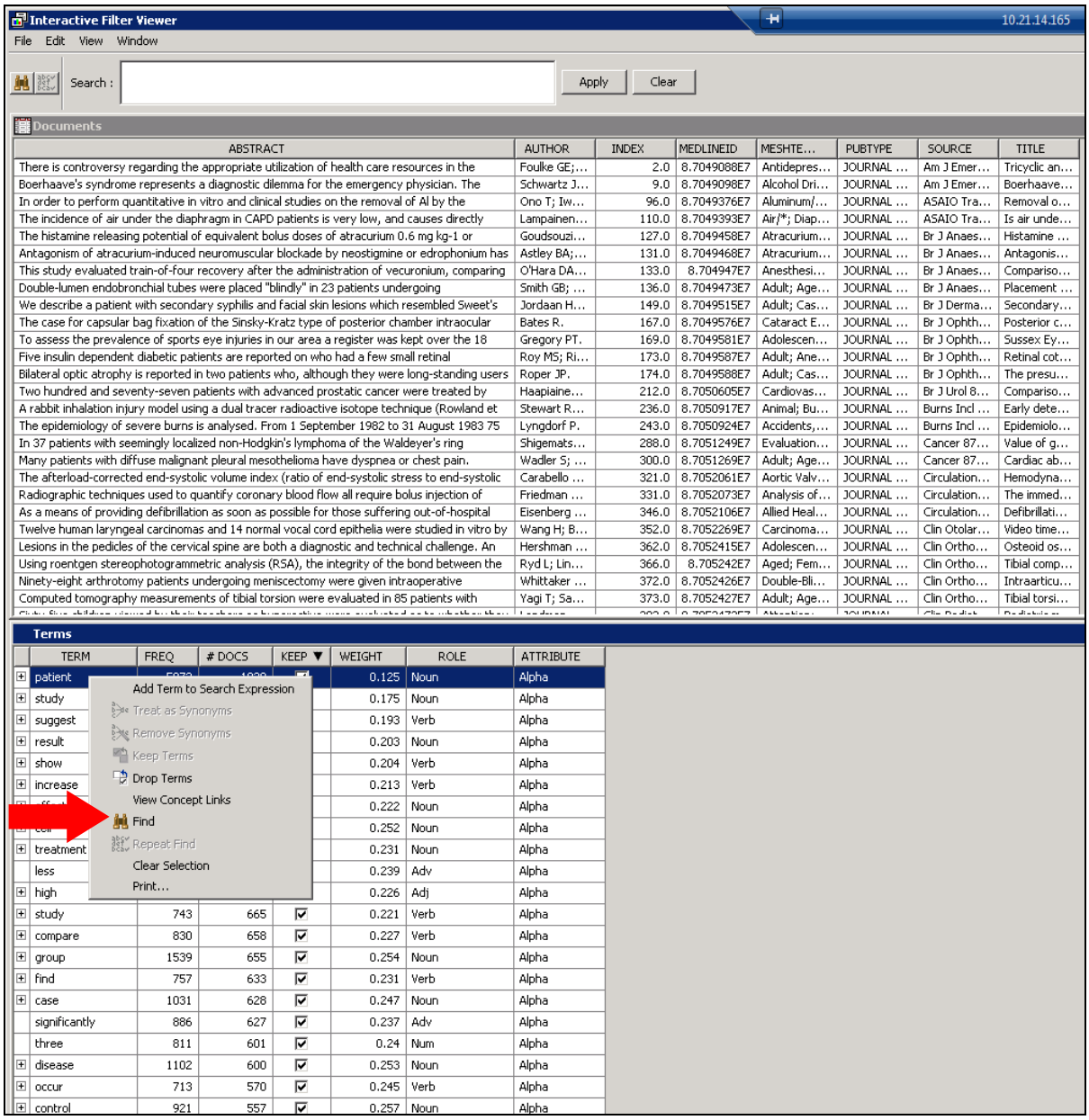
3. Attach a **Text Filter** node to the **Text Parsing** node. The default frequency weighting is **Log**. When there is no target variable, the default term weight is **Entropy**. It is a good idea to make this explicit, as shown.



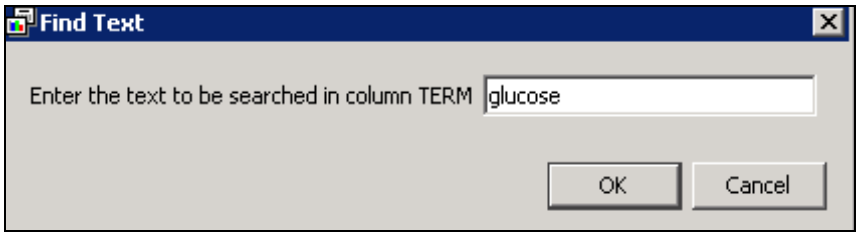
4. Run the default Text Parsing and Text Filter nodes.

5. Select **Filter Viewer** from the Properties Panel. This accesses the Interactive Filter Viewer where searching on terms in the documents is performed.

6. In the Terms window, right-click any term in the table. Select **Find**.



7. Enter **glucose** as the term to find.

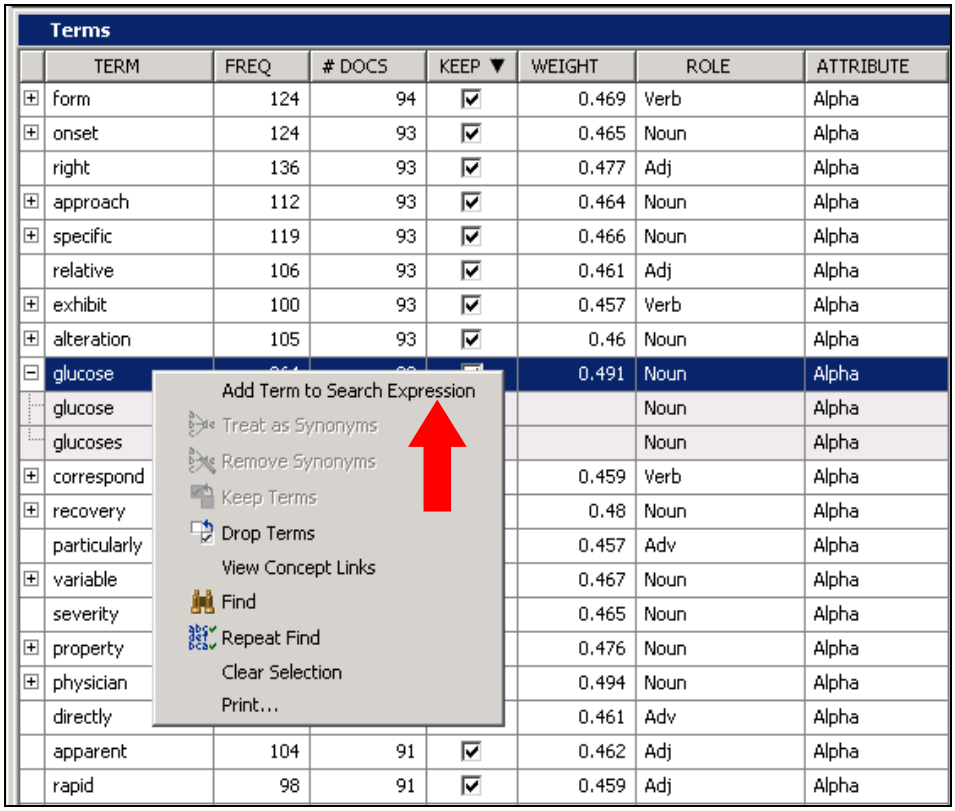


The table jumps to the portion of the table that contains the term **glucose**.

⊟	glucose	264	93	☑	0.491	Noun	Alpha
	glucose	263	93			Noun	Alpha
	glucoses	1	1			Noun	Alpha

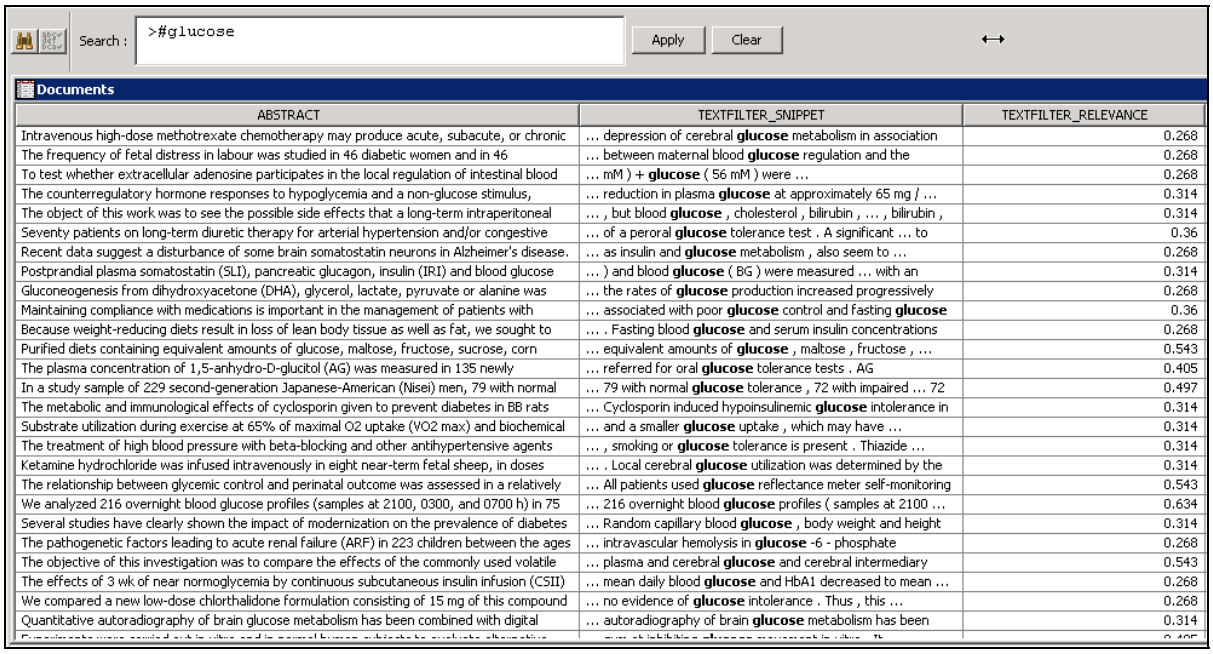
Expand ⊞ to see the stemmed versions of **glucose**. It occurred 263 times in its singular form and one time as a plural.

8. Right-click on the first row of **glucose** and select **Add Term to Search Expression**.



The term **glucose** is added to the Search window. The preceding symbols (>#) indicate that all stemmed versions (or synonyms if any were defined) of the term are searched for.

9. Click **Apply**. The following results appear in the **Documents** window:

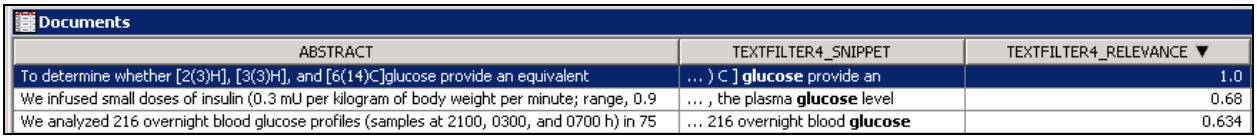


ABSTRACT	TEXTFILTER_SNIPPET	TEXTFILTER_RELEVANCE
Intravenous high-dose methotrexate chemotherapy may produce acute, subacute, or chronic	... depression of cerebral **glucose** metabolism in association	0.268
The frequency of fetal distress in labour was studied in 46 diabetic women and in 46	... between maternal blood **glucose** regulation and the	0.268
To test whether extracellular adenosine participates in the local regulation of intestinal blood	... mM) + **glucose** (56 mM) were ...	0.268
The counterregulatory hormone responses to hypoglycemia and a non-glucose stimulus,	... reduction in plasma **glucose** at approximately 65 mg / ...	0.314
The object of this work was to see the possible side effects that a long-term intraperitoneal	... , but blood **glucose** , cholesterol , bilirubin , ... , bilirubin ,	0.314
Seventy patients on long-term diuretic therapy for arterial hypertension and/or congestive	... of a peroral **glucose** tolerance test . A significant ... to	0.36
Recent data suggest a disturbance of some brain somatostatin neurons in Alzheimer's disease.	... as insulin and **glucose** metabolism , also seem to ...	0.268
Postprandial plasma somatostatin (SLI), pancreatic glucagon, insulin (IRI) and blood glucose	...) and blood **glucose** (BG) were measured ... with an	0.314
Gluconeogenesis from dihydroxyacetone (DHA), glycerol, lactate, pyruvate or alanine was	... the rates of **glucose** production increased progressively	0.268
Maintaining compliance with medications is important in the management of patients with	... associated with poor **glucose** control and fasting **glucose**	0.36
Because weight-reducing diets result in loss of lean body tissue as well as fat, we sought to	... . Fasting blood **glucose** and serum insulin concentrations	0.268
Purified diets containing equivalent amounts of glucose, maltose, fructose, sucrose, corn	... equivalent amounts of **glucose** , maltose , fructose , ...	0.543
The plasma concentration of 1,5-anhydro-D-glucitol (AG) was measured in 135 newly	... referred for oral **glucose** tolerance tests . AG	0.405
In a study sample of 229 second-generation Japanese-American (Nisei) men, 79 with normal	... 79 with normal **glucose** tolerance , 72 with impaired ... 72	0.497
The metabolic and immunological effects of cyclosporin given to prevent diabetes in BB rats	... Cyclosporin induced hypoinsulinemic **glucose** intolerance in	0.314
Substrate utilization during exercise at 65% of maximal O2 uptake (VO2 max) and biochemical	... and a smaller **glucose** uptake , which may have ...	0.314
The treatment of high blood pressure with beta-blocking and other antihypertensive agents	... , smoking or **glucose** tolerance is present . Thiazide ...	0.314
Ketamine hydrochloride was infused intravenously in eight near-term fetal sheep, in doses	... . Local cerebral **glucose** utilization was determined by the	0.314
The relationship between glycemic control and perinatal outcome was assessed in a relatively	... All patients used **glucose** reflectance meter self-monitoring	0.543
We analyzed 216 overnight blood glucose profiles (samples at 2100, 0300, and 0700 h) in 75	... 216 overnight blood **glucose** profiles (samples at 2100 ...	0.634
Several studies have clearly shown the impact of modernization on the prevalence of diabetes	... Random capillary blood **glucose** , body weight and height	0.314
The pathogenetic factors leading to acute renal failure (ARF) in 223 children between the ages	... intravascular hemolysis in **glucose** -6 - phosphate	0.268
The objective of this investigation was to compare the effects of the commonly used volatile	... plasma and cerebral **glucose** and cerebral intermediary	0.543
The effects of 3 wk of near normoglycemia by continuous subcutaneous insulin infusion (CSII)	... mean daily blood **glucose** and HbA1 decreased to mean ...	0.268
We compared a new low-dose chlorthalidone formulation consisting of 15 mg of this compound	... no evidence of **glucose** intolerance . Thus , this ...	0.268
Quantitative autoradiography of brain glucose metabolism has been combined with digital	... autoradiography of brain **glucose** metabolism has been	0.314

The abstract is shown on the left. Stretch the column labeled **TEXTFILTER_SNIPPET** so that you can see the term **glucose** in every row. This indicates the part of the abstract where **glucose** appears. (This is the first occurrence if there are multiple occurrences.)

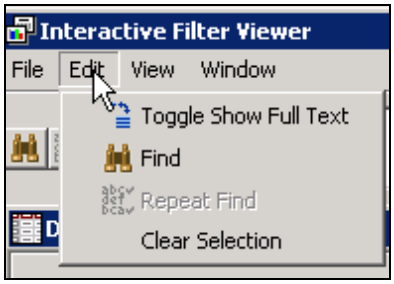
Place your mouse pointer above the **TEXTFILTER_SNIPPET** label. You see the following message: “Left-click on column header to sort 93 rows of the table.” This indicates that 93 documents were selected because either **glucose** or **glucoses** (or both) are found at least once in each document.

The **TEXTFILTER_RELEVANCE** column returns a measure of how strongly each document is associated with the search term. This is a relative measure. The most relevant document is given the highest value of 1. The calculation of this metric considers factors such as the number of times a term (or its stemmed versions and synonyms) appears in a document. To get an idea of this, click twice on the column heading for **TEXTFILTER_RELEVANCE** until you see the most relevant document in the first row (the one with **TEXTFILTER_RELEVANCE**=1.0). Then select that row.



ABSTRACT	TEXTFILTER4_SNIPPET	TEXTFILTER4_RELEVANCE ▼
To determine whether [2(3)H], [3(3)H], and [6(14)C]glucose provide an equivalent	...) C] **glucose** provide an	1.0
We infused small doses of insulin (0.3 mU per kilogram of body weight per minute; range, 0.9	... , the plasma **glucose** level	0.68
We analyzed 216 overnight blood glucose profiles (samples at 2100, 0300, and 0700 h) in 75	... 216 overnight blood **glucose**	0.634

Select **Edit** ⇨ **Toggle Show Full Text** to see the complete document with the highest relevance score.

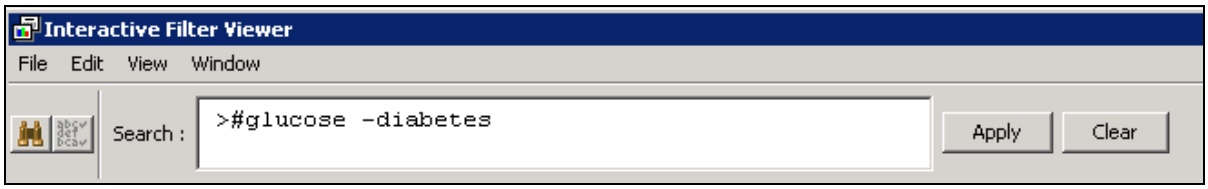


The full text for this abstract can be read.

ABSTRACT	TEXTFILTER4_SNIPPET
To determine whether [2(3)H], [3(3)H], and [6(14)C]glucose provide an equivalent assessment of glucose turnover in insulin-dependent diabetes mellitus (IDDM) and nondiabetic man, glucose utilization rates were measured using a simultaneous infusion of these isotopes before and during hyperinsulinemic euglycemic clamps. In the nondiabetic subjects, glucose turnover rates determined with [6(14)C]glucose during insulin infusion were lower (P less than 0.02) than those determined with [2(3)H]glucose and higher (P less than 0.01) than those determined with [3(3)H]glucose. In IDDM, glucose turnover rates measured with [6(14)C]glucose during insulin infusion were lower (P less than 0.05) than those determined with [2(3)H]glucose, but were not different from those determined with [3(3)H]glucose. All three isotopes indicated the presence of insulin resistance. However, using [3(3)H]glucose led to the erroneous conclusion that glucose utilization was not significantly decreased at high insulin concentrations in the diabetic patients. [6(14)C] and [3(3)H]glucose but not [2(3)H]glucose indicated impairment in insulin-induced suppression of glucose production. These results indicate that tritiated isotopes do not necessarily equally reflect the pattern of glucose metabolism in diabetic and nondiabetic man.	...) C] **glucose** provide an equivalent assessment of ... equivalent assessment of **glucose** turnover in insulin-dependent diabetes mellitus ... nondiabetic man , **glucose** utilization rates were measured using ...

Reading through the full document, it is obvious that **glucose** is used many times. This explains why this document has the highest relevance measure for a query based on this term. Select **Edit** ⇨ **Toggle Show Full Text** to go back to the original way of viewing the documents.

10. Ninety-three documents were retrieved by the query. It is also useful to be able to retrieve documents that contain one term, but do ***not*** contain another term. For example, in order to take these 93 documents and eliminate any that contain the term **diabetes**, in the Search window, enter **–diabetes**. (That is, precede the term with a minus sign as shown below.) Select **Apply**.



Be sure to separate the two terms with a space.

Verify that 72 documents of the original 93 remain after you eliminate any documents with the term **diabetes**.

Exercises

2. Finding a Text String in a Movies Data Set

This simple exercise searches through documents using the Filter Viewer.

a. Create a new diagram called **Movies**.

Into this diagram, bring the data table stored as **DMTXT.MOVIESGENRE** (**label=Movies Data with Genre Codes**). This contains 1,527 movie synopses randomly selected from movie descriptions, reviews, and summaries found on the Internet. Some movies might have multiple entries.

b. Connect the default **Text Parsing** and **Text Filter** nodes to this data set.

c. Identify the name of an actor or actress of interest to you. Using the Interactive Filter Viewer of the Text Filter node, find all of the movies in the data set that have a synopsis that mentions the name that you selected.

d. Has Brad Pitt ever portrayed a vampire in a movie?

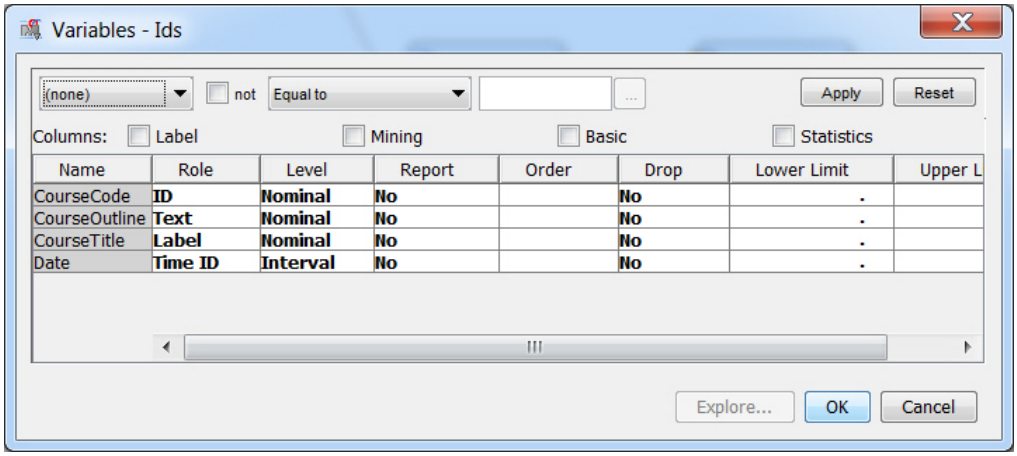
3. Text Mining SAS Course Descriptions (Combination Exercise and Self-Guided Demonstration)

This exercise is intended to show how useful information retrieval and text mining can be for activities such as call routing. In this example, there is no target variable and the emphasis is on information retrieval and document categorization.

SAS Education supports more than 300 courses. Prospective students often have questions about curriculum and specific course content. For example, a prospect might ask for information about courses that discuss neural networks. Text mining provides a solution for automating queries based on keywords.

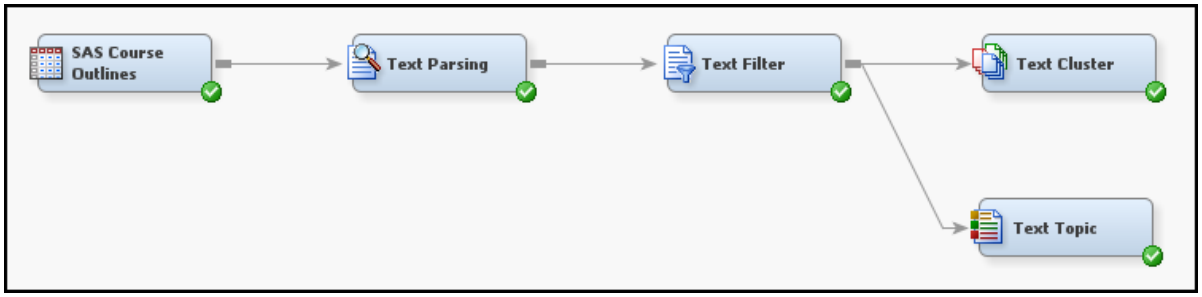
Descriptions of the SAS courses can be found at **http://support.sas.com/training**.

The SAS course descriptions data set **DMTXT.SASCOURSES** contains descriptions of courses supported in 2011. The data set has 735 rows (documents) and four columns. Some courses have multiple versions that are associated with different releases of the software. The metadata is shown below.



The variable **CourseOutline** contains the course outline text.

The following flow diagram implements the text mining analysis:



Frequency filtering is a methodology to create or add to a stop list. You can run the Text Parsing node with the default stop list and then use frequency filtering to add terms to this list. Frequency filtering specifies a cutoff frequency. Terms with a frequency below the cutoff are added to the stop list. You can also specify a cutoff frequency at the high end so that terms with a frequency above the cutoff are added to the stop list. For creating a start list, keep terms with frequencies between the high and low cutoff values. The start list **DMTXT.SASCOURSESTART** contains a start list that was obtained using domain knowledge and frequency filtering.

a. Create a diagram named **SAS Course Outlines**. If you need to, create a data source for the **DMTXT.SASCOURSES** data set. (The metadata is presented above.) Drag this data source into the diagram.

b. Attach a **Text Parsing** node to the **Input Data Source** node. Change the Synonyms property so that there is no synonyms table and add **DMTXT.SASCOURSESTART** as a start list.

c. Attach a **Text Filter** node to the **Text Parsing** node. Change frequency weighting from Default to **Log**. Change term weighting from Default to **Inverse Document Frequency**. **Log** is the default frequency weight, but **Entropy** is the default term weight. **Inverse Document Frequency** is recommended for documents larger than a paragraph. Run the Text Filter node.

d. Open the Filter Viewer (also called the *Interactive Filter Viewer*). Determine how many documents contain the term *neural network* by doing a search on this term. How many documents did your search return? Why is this number not 23?

e. Select the document corresponding to the course with the code BASEL52. Select **Edit** ⇨ **Toggle Show Full Text**. You can read the course outline for BASEL52.

f. Select **Clear** ⇨ **Apply** to return all of the documents in the collection. Navigate back to the **neural network** row in the Terms table. Right-click on the neural network **TERM** cell and select **View Concept Links**. The concept link plot appears. What are some of the terms strongly associated with *neural network*?

g. Close the Filter Viewer. Attach a **Text Topic** node to the **Text Filter** node. For User Topics in the Property Sheet, select the data set **DMTXT.SASTOPICS**. Keep all of the other default settings. Run the Text Topic node.

h. Access the Results window. Which topic contains the most documents?

i. Close the Results window for the Text Topic node. Look at the exported data and determine what variables were created by this node.

j. Open the Topic Viewer from the Property Sheet and explore the results.

A custom topic is similar to a predefined query. The topic weight shown in the documents window determines whether the topic is present. (That is, the query is satisfied.) If the topic weight exceeds the document cutoff, then the document is classified as having the topic.

k. Close the Topic Viewer. Attach a **Text Cluster** node to the **Text Filter** node. Most users attach the Text Topic node and Text Cluster node directly to each other, but they work independently. Neither requires any results from the other.

l. Use the default setting and run the **Text Cluster** node. Open the Results window. How many clusters were created? Can you interpret some of the clusters from the displayed descriptive terms? How many SDV variables were created?

2.4 Chapter Summary

SAS provides many tools and products for accessing and processing data. The SAS language features a rich set of character functions for processing text. In addition, Perl regular expressions are supported. The most typical way to bring in text documents for SAS Text Miner is to store them as individual files in a directory and use the Text Import node to convert them to a SAS data set)one row per document). The original documents can be in many different formats (PDF, RTF, Word, PPT, and so on). If the document files are saved with a name that corresponds to a key ID variable, simple data processing can then be used to match the documents to other variables contained in a separate data set.

When there is a target variable, it is possible to assess how much derived text variables incrementally add predictive power to other input variables.

Analytic techniques for text analysis can be used to identify authorship. This has applications in historical research and can also contribute to forensic linguistics.

Information retrieval (IR) methods are designed to access relevant information quickly. The Interactive Filter Viewer in the Text Filter node supports queries to extract relevant documents.

For Additional Information

Swanson, Don R. 1988. “Migraine and Magnesium - 11 Neglected Connections.” *Perspectives in Biology and Medicine*. 31 (4). pp. 526-557.

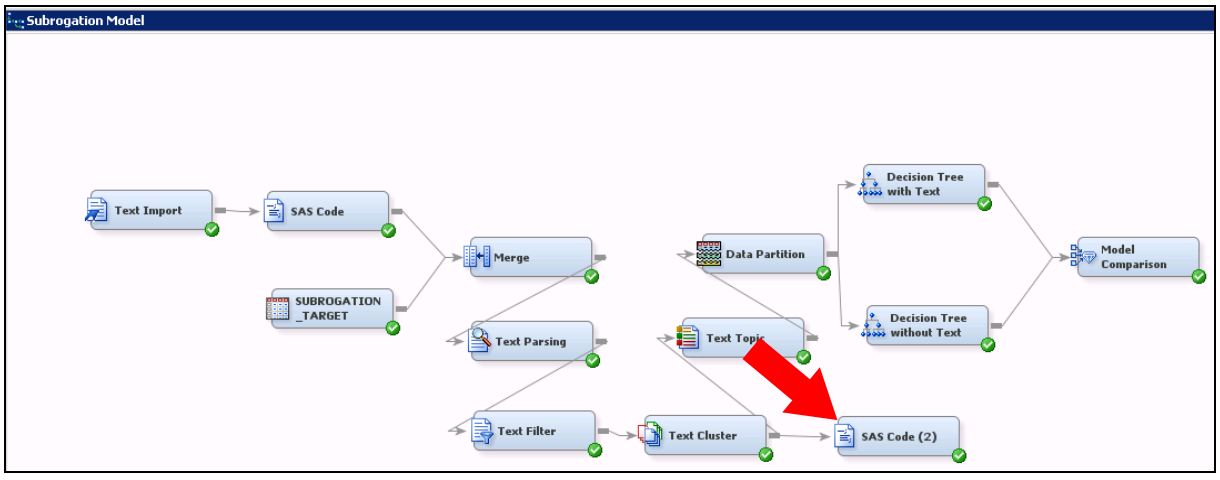
Swanson, Don R. 1991. “Complementary structures in disjoint science literatures.” *SIGIR’91. Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*. pp. 280-289.

2.5 Solutions

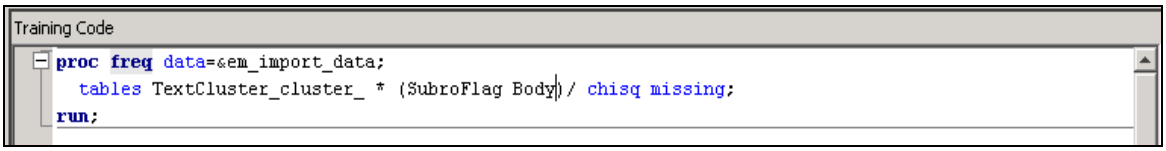
Solutions to Exercises

1. **Interpreting the Document Clusters in Relation to Other Variables**

   a. The SAS Code node is attached as below. The PROC FREQ procedure is set up in the Training Code window.



Set up a PROC FREQ procedure in the following way:



The CHISQ option shows the Pearson chi-square test for the independence of rows and columns. The MISSING option creates a table that shows any missing values. (These options are not needed for this exercise.)

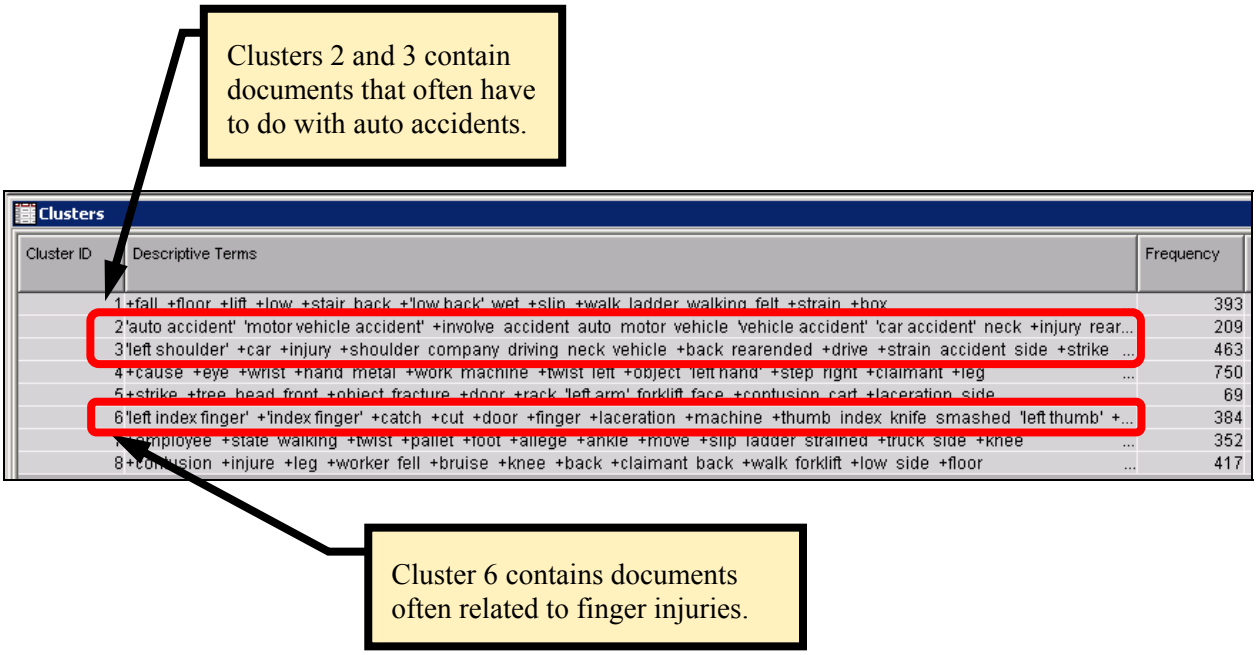
b. Look at the results for the **TextCluster_cluster_*SubroFlag** crosstabulation. Cluster 2 has a 87.08% subrogation success rate and Cluster 3 has a 75.16% success rate. These are the two clusters with the best rates. The descriptive terms for these two clusters indicate that the documents in these two clusters are often characterized by some type of automobile accident.

TextCluster_cluster_ SubroFlag(Subrogation (1=Yes 0=No))

Frequency Percent Row Pct Col Pct	0	1	Total
1	306 10.08 77.86 16.09	87 2.86 22.14 7.67	393 12.94
2	27 0.89 12.92 1.42	182 5.99 87.08 16.04	209 6.88
3	115 3.79 24.84 6.05	348 11.46 75.16 30.66	463 15.25
4	564 18.57 75.20 29.65	186 6.12 24.80 16.39	750 24.70
5	43 1.42 62.32 2.26	26 0.86 37.68 2.29	69 2.27
6	344 11.33 89.58 18.09	40 1.32 10.42 3.52	384 12.64
7	234 7.70 66.48 12.30	118 3.89 33.52 10.40	352 11.59
8	269 8.86 64.51 14.14	148 4.87 35.49 13.04	417 13.73
Total	1902 62.63	1135 37.37	3037 100.00

Clusters 2 and 3 have the highest subrogation success rates.

Worst subrogation rate is 10.42% (cluster 6).



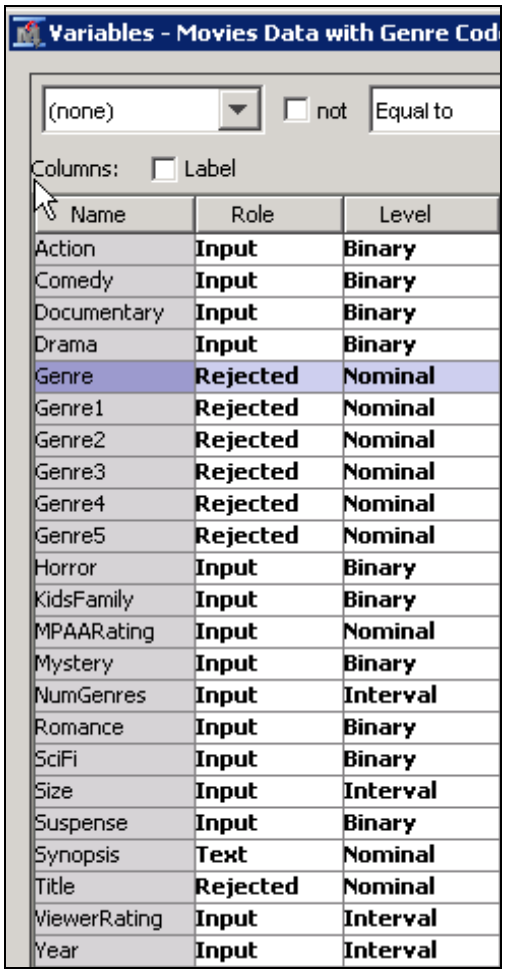
c. Cluster 6 contains the Adjustor notes ***least*** likely to lead to a successful subrogation. Only 10.42% of these claims were successes.

The Adjustor Note with the sentence "Claimant trying to pry pallet jack from in between dock and trailer where it was stuck, slipped and smashed finger in between the two." indicates a finger injury. This note is in Cluster 6, which has the lowest probability of successful subrogation. This cluster has only a 10.42% success rate.

2. Finding a Text String in a Movies Data Set

Set up the flow diagram in the standard way.

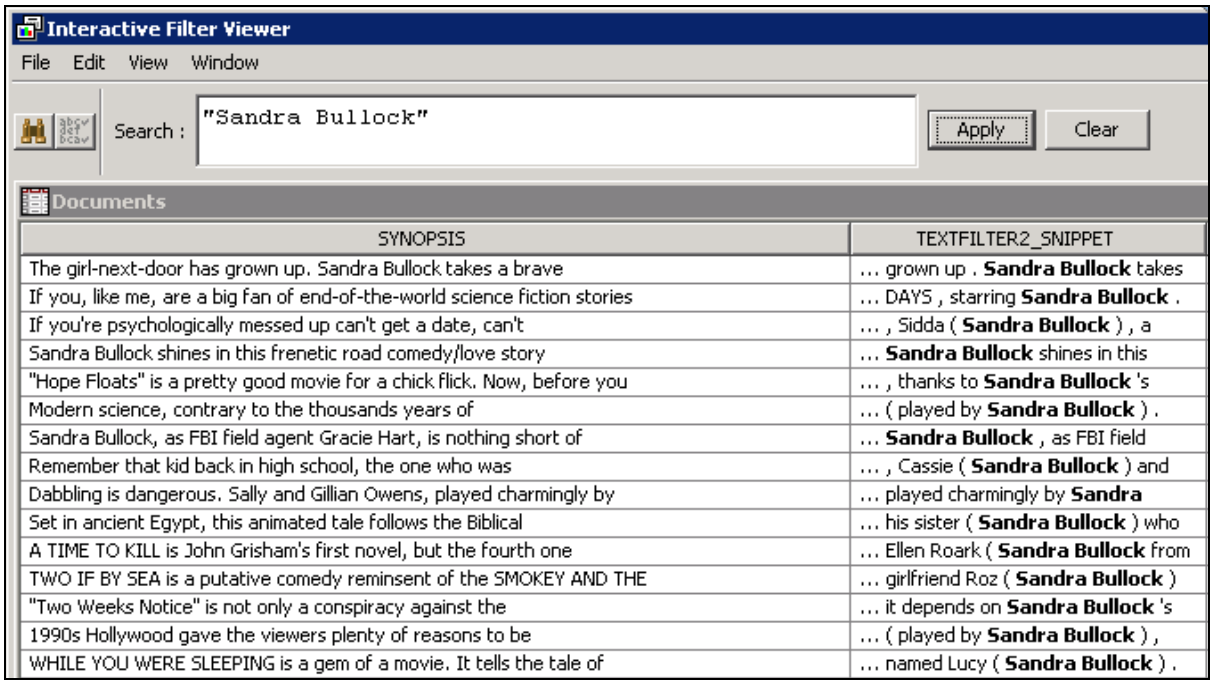
The data set was created with the following variable definitions:



Name	Role	Level
Action	**Input**	**Binary**
Comedy	**Input**	**Binary**
Documentary	**Input**	**Binary**
Drama	**Input**	**Binary**
Genre	**Rejected**	**Nominal**
Genre1	**Rejected**	**Nominal**
Genre2	**Rejected**	**Nominal**
Genre3	**Rejected**	**Nominal**
Genre4	**Rejected**	**Nominal**
Genre5	**Rejected**	**Nominal**
Horror	**Input**	**Binary**
KidsFamily	**Input**	**Binary**
MPAARating	**Input**	**Nominal**
Mystery	**Input**	**Binary**
NumGenres	**Input**	**Interval**
Romance	**Input**	**Binary**
SciFi	**Input**	**Binary**
Size	**Input**	**Interval**
Suspense	**Input**	**Binary**
Synopsis	**Text**	**Nominal**
Title	**Rejected**	**Nominal**
ViewerRating	**Input**	**Interval**
Year	**Input**	**Interval**

Synopsis is the Text variable to analyze. Run the nodes. (This takes a few minutes to do.)

Go to the **Interactive Filter Viewer**. Suppose you are interested in finding movies that star Sandra Bullock. There are at least two ways to search on her name. One way is simply to enter **"Sandra Bullock"** (with quotation marks, but upper or lower case makes no difference) in the Search window. This retrieves 15 documents.



SYNOPSIS	TEXTFILTER2_SNIPPET
The girl-next-door has grown up. Sandra Bullock takes a brave	... grown up . **Sandra Bullock** takes
If you, like me, are a big fan of end-of-the-world science fiction stories	... DAYS , starring **Sandra Bullock** .
If you're psychologically messed up can't get a date, can't	... , Sidda (**Sandra Bullock**) , a
Sandra Bullock shines in this frenetic road comedy/love story	... **Sandra Bullock** shines in this
"Hope Floats" is a pretty good movie for a chick flick. Now, before you	... , thanks to **Sandra Bullock** 's
Modern science, contrary to the thousands years of	... (played by **Sandra Bullock**) .
Sandra Bullock, as FBI field agent Gracie Hart, is nothing short of	... **Sandra Bullock** , as FBI field
Remember that kid back in high school, the one who was	... , Cassie (**Sandra Bullock**) and
Dabbling is dangerous. Sally and Gillian Owens, played charmingly by	... played charmingly by **Sandra**
Set in ancient Egypt, this animated tale follows the Biblical	... his sister (**Sandra Bullock**) who
A TIME TO KILL is John Grisham's first novel, but the fourth one	... Ellen Roark (**Sandra Bullock** from
TWO IF BY SEA is a putative comedy reminsent of the SMOKEY AND THE	... girlfriend Roz (**Sandra Bullock**)
"Two Weeks Notice" is not only a conspiracy against the	... it depends on **Sandra Bullock** 's
1990s Hollywood gave the viewers plenty of reasons to be	... (played by **Sandra Bullock**) ,
WHILE YOU WERE SLEEPING is a gem of a movie. It tells the tale of	... named Lucy (**Sandra Bullock**) .

Another search expression is to use the following:

This also retrieves the same 15 documents.

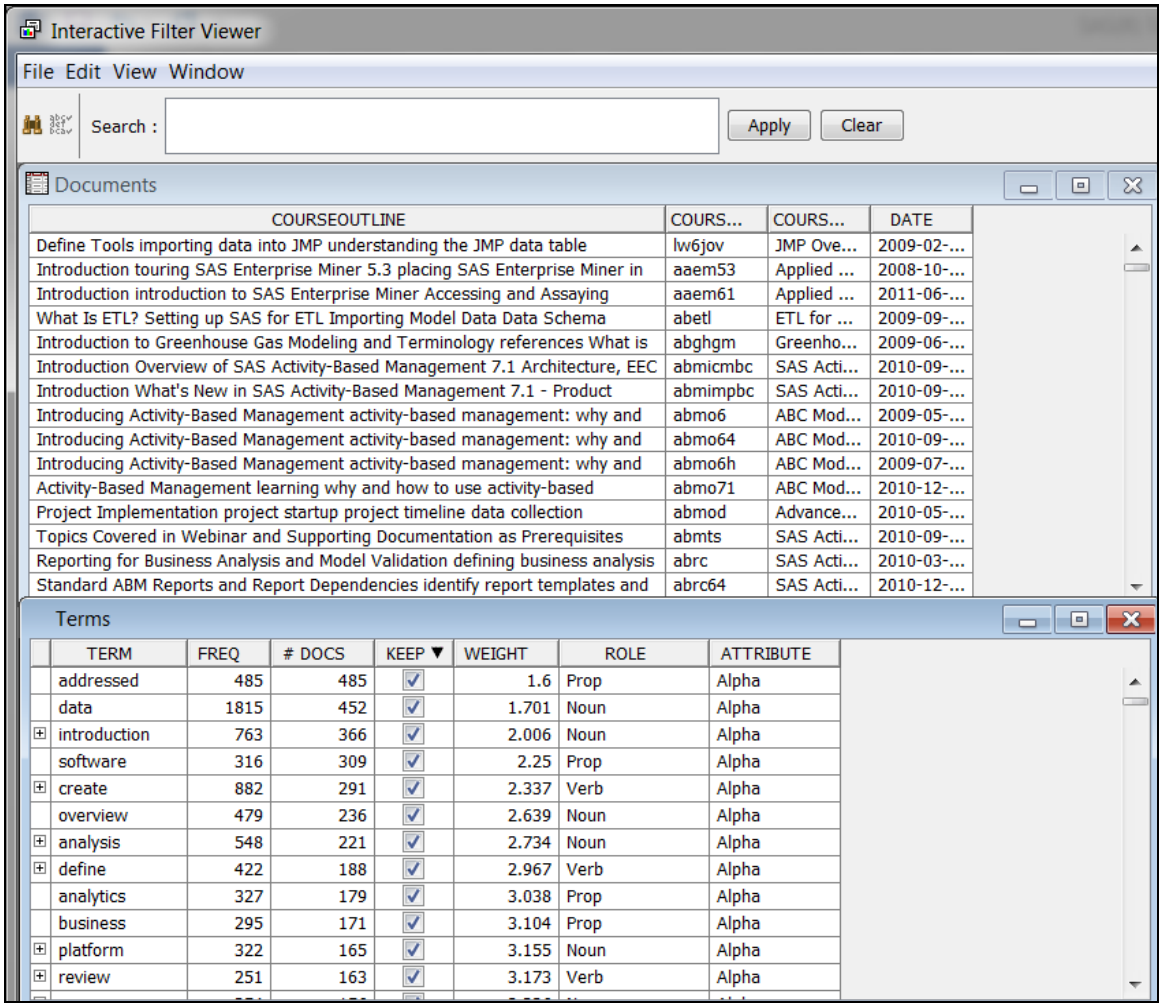
As to whether Brad Pitt ever played in a vampire movie, a search on **Brad Pitt** (or **+brad +pitt**) returns 16 documents. If you scroll to the right and look at the title of each of these movies, you see that he indeed played a vampire. (You can also read the Synopsis for this movie to verify it.) Did you see this movie?

TITLE
Abandon
Being John Malkovich
Confessions Of A Dangerous Mind
Cool World
Fight Club
Interview with the Vampire
Kundun
Legends of the Fall
Meet Joe Black
Mexican, The
Ocean's Eleven
Seven
Seven Years in Tibet
Sleepers
Snatch
Troy

3. Text Mining SAS Course Descriptions (Combination Exercise and Self-Guided Demonstration)

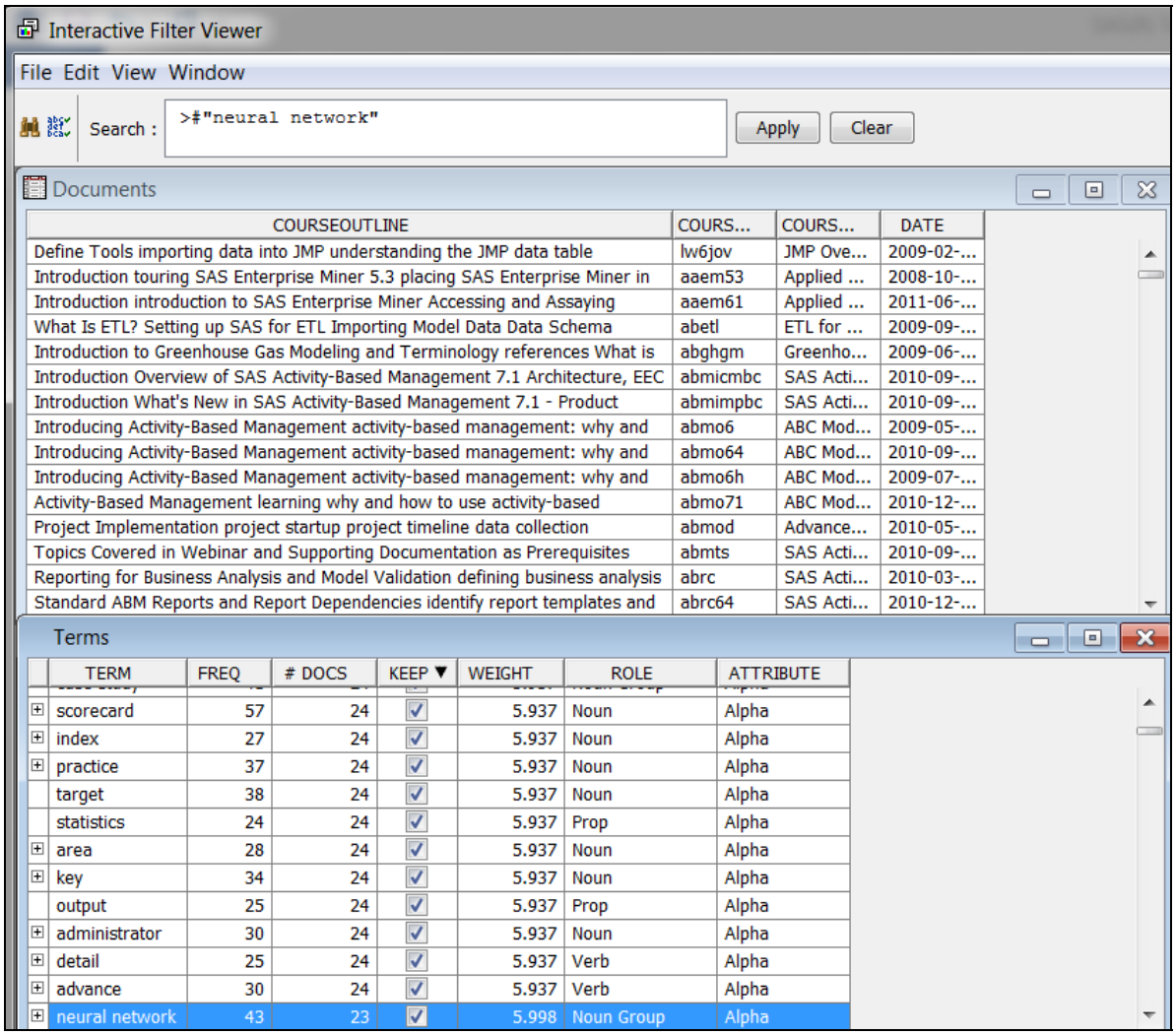
The following walks you through the major steps of this exercise and demo. It also introduces you to some features that were not previously mentioned.

After you set up the diagram and run the Text Filter node as specified earlier, the Interactive Filter Viewer should resemble the following:

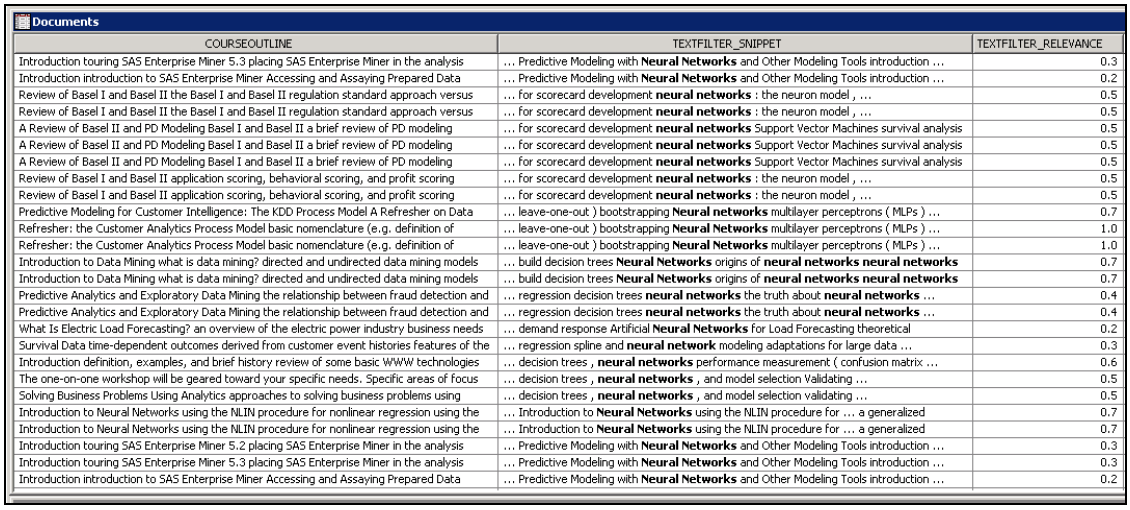


If you sort the TERM column in the Terms table by clicking on the heading cell containing the word **TERM**, then you can use a quick-find feature. Select any term in the TERM column. Then enter the first letter of the term that you want to find. The window is scrolled to the first term starting with that letter. You can also select **Edit** ⇨ **Find** to go directly to a desired term.

In the Filter Viewer, select **Edit** ⇨ **Find**, and enter the two word phrase **neural network**. After you click **OK**, you are taken to the first cell in the TERM column that contains the phrase *neural network*. Notice that there are 23 documents containing this phrase. Right-click in the cell containing the phrase *neural network*, and select **Add Term to Search Expression**. The Search window contains **>#"neural network"**. The quotation marks are required if the search expression has more than one word. If you look at the filter rules, you see that this expression searches for documents containing *neural network* and any of the synonyms of *neural network*.



Click **Apply**. The following results appear in the Documents window:



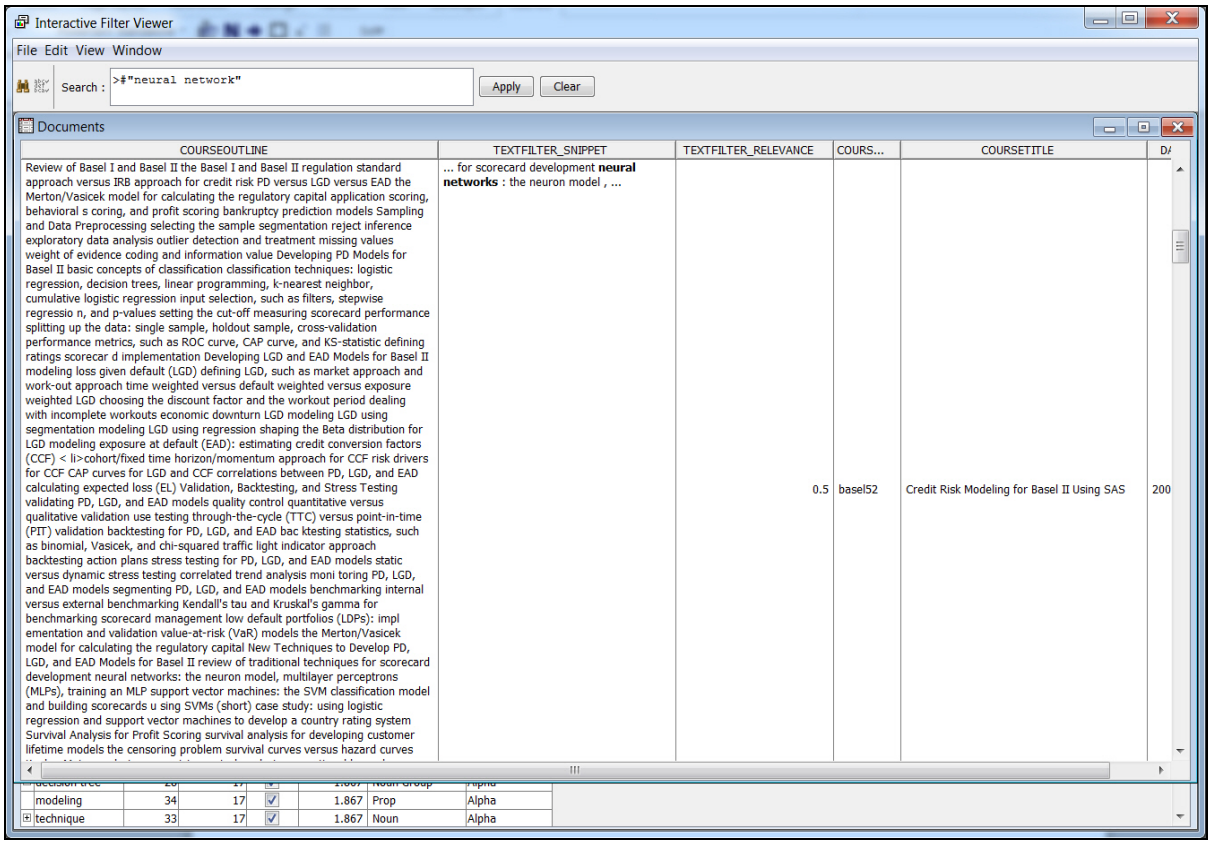
Stretch the **TEXTFILTER_SNIPPET** column so that you can see the term *neural networks* for all the listed documents. The **TEXTFILTER_RELEVANCE** column contains the result of the inner product calculation described in the previous discussion of a Boolean query. The cutoff is not displayed, but all documents that produced a result above the cutoff are returned. Typically, the courses with lower relevance scores include neural network material, but the courses include other material as well. The neural network portion is a small section of the course. You can determine that 31 documents were returned by placing your cursor over any of the column headings.

On the other hand, when you look at the Terms window, you see that *neural network* appears sometimes as a noun group and sometimes as a simple noun. ***These are treated as two separate types of terms.*** Consequently, the number of documents in which these two types of terms appear (23+12=35) does not have to agree with 31 from above. A single document could contain *neural network* at least once as a noun group and then *neural network* appears elsewhere in the document as a noun.

Terms

TERM ▲	FREQ	# DOCS	KEEP	WEIGHT	ROLE	ATTRIBUTE
⊞ network data	2	2	☐	0.0	Noun Group	Alpha
network flow	0	0	☐	0.0	Noun Group	Alpha
⊞ network neural ne…	2	2	☐	0.0	Noun Group	Alpha
⊞ network preference	0	0	☐	0.0	Noun Group	Alpha
network-info	0	0	☐	0.0	Noun	Mixed
networked	2	2	☐	0.0	Prop	Alpha
networking	0	0	☐	0.0	Noun	Alpha
networks	5	5	☐	0.0	Prop	Alpha
neural	38	21	☑	1.562	Adj	Alpha
neural	11	11	☐	0.0	Prop	Alpha
⊟ neural network	43	23	☑	1.431	Noun Group	Alpha
neural network	11	11			Noun Group	Alpha
neural networks	32	21			Noun Group	Alpha
neural network	12	12	☐	0.0	Noun	Alpha

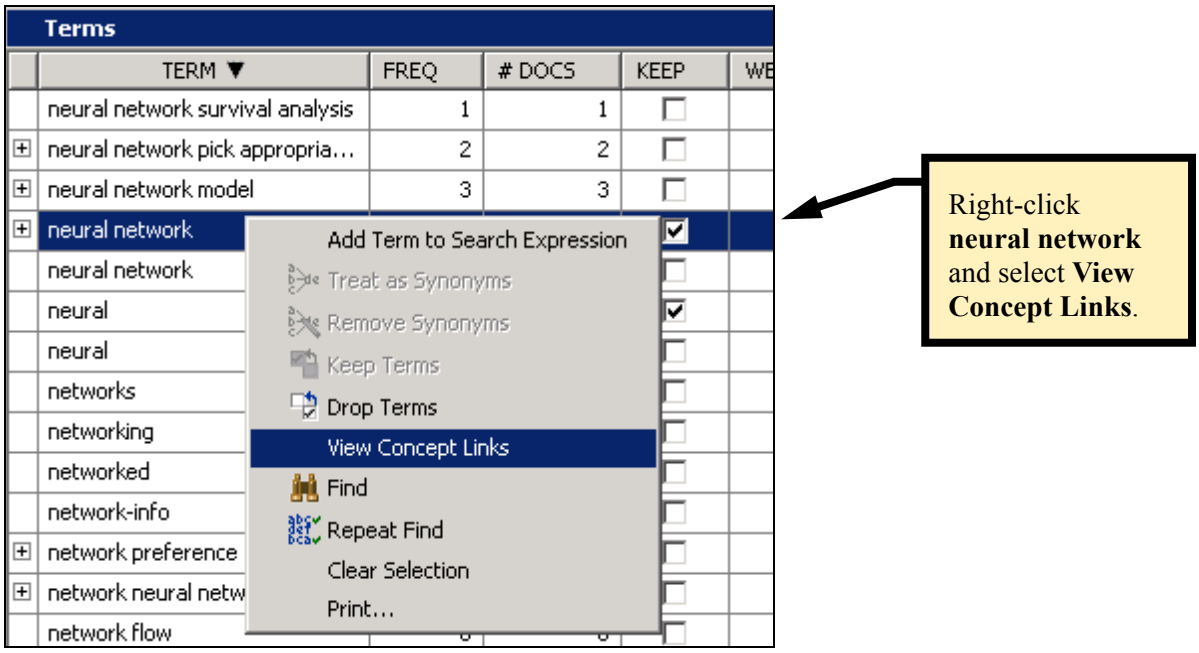
You can also look at a full document. Select the document corresponding to the course with the code BASEL52. Select **Edit** ⇨ **Toggle Show Full Text**. You can read the course outline for BASEL52.



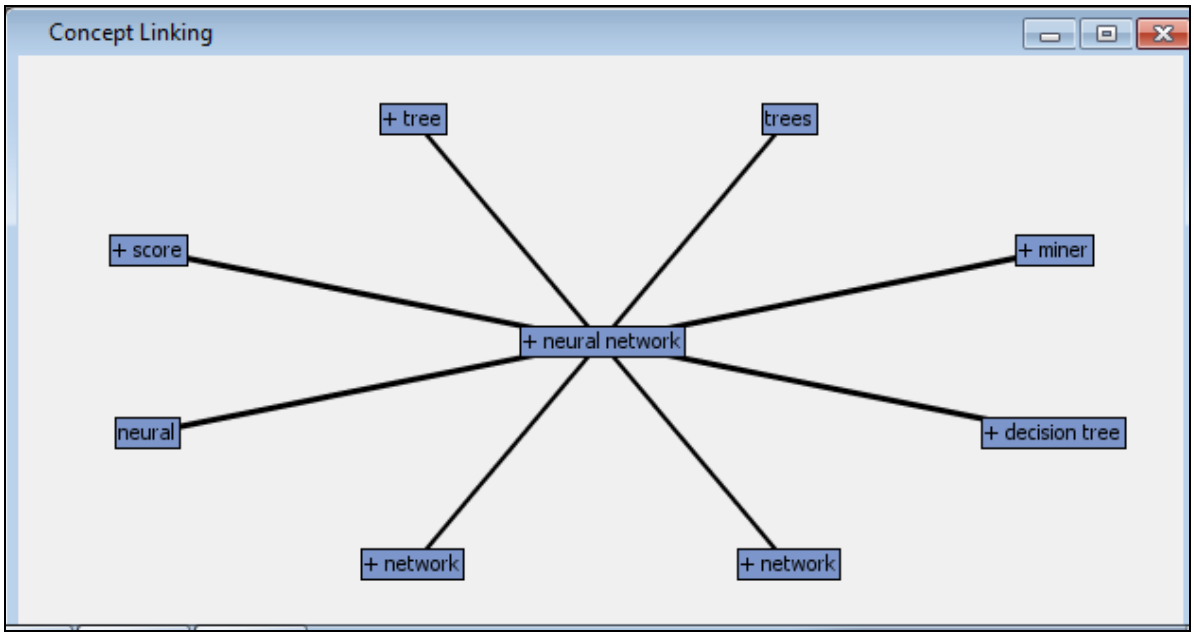
The relevance score for this document (.5) is in the middle of the relevance values for returned documents. It is not an extremely high value because most of the course outline discusses topics that are not related to neural networks. However, neural networks and terms that tend to appear in relation to neural networks (for example, KS-statistic and ROC curve) do appear in this document. You can select **Edit** ⇨ **Toggle Show Full Text** again to return to one row per document.

Of the 31 returned documents, most appear to be legitimately related to neural networks, so the precision of this query (percent of the documents returned that are relevant to the search query) is approximately 100%.

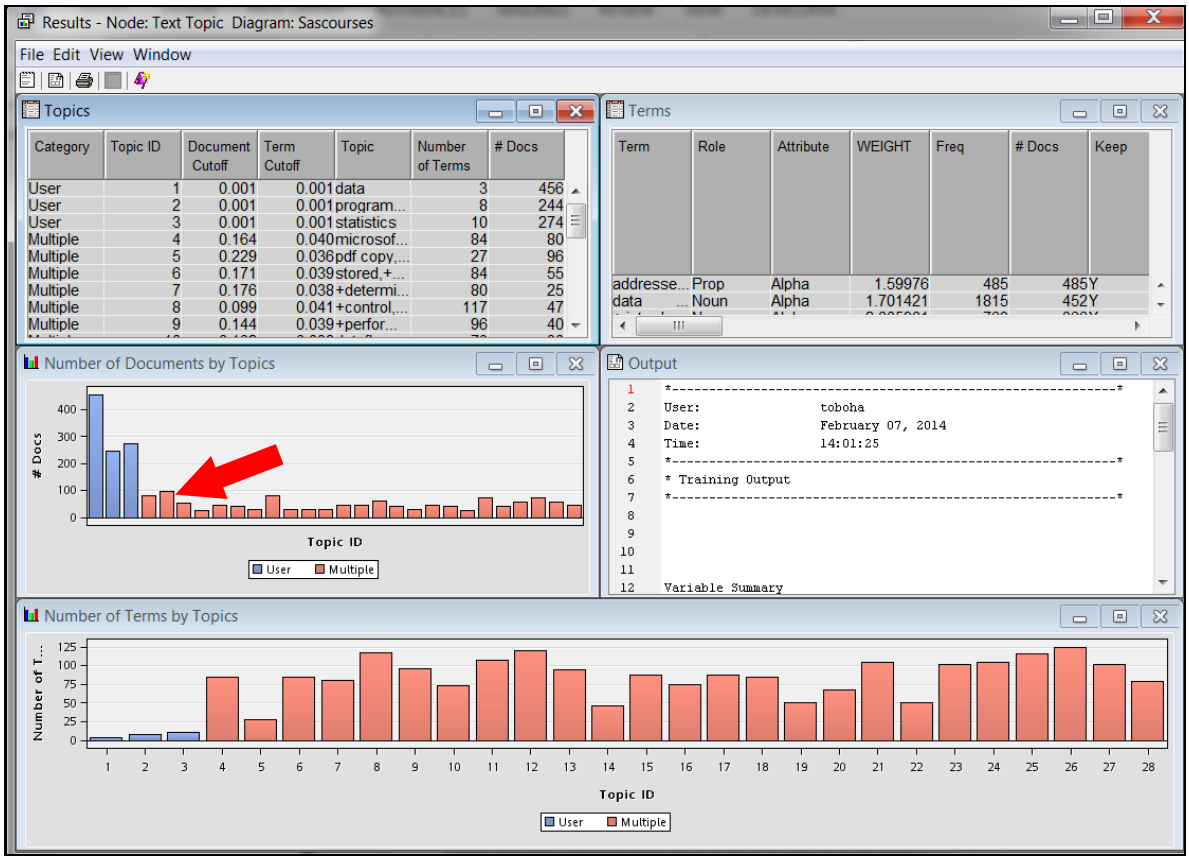
Select **Clear** ⇨ **Apply** to retrieve all of the documents in the collection. Navigate again to the neural network row in the Terms table. Right-click the **neural network** cell. select **View Concept Links**. The concept link plot appears.



The Concept Linking window appears. You can see the terms most strongly associated with *neural network*. You can also right-click on any of these terms and select **Expand Links** to look at indirectly associated terms.



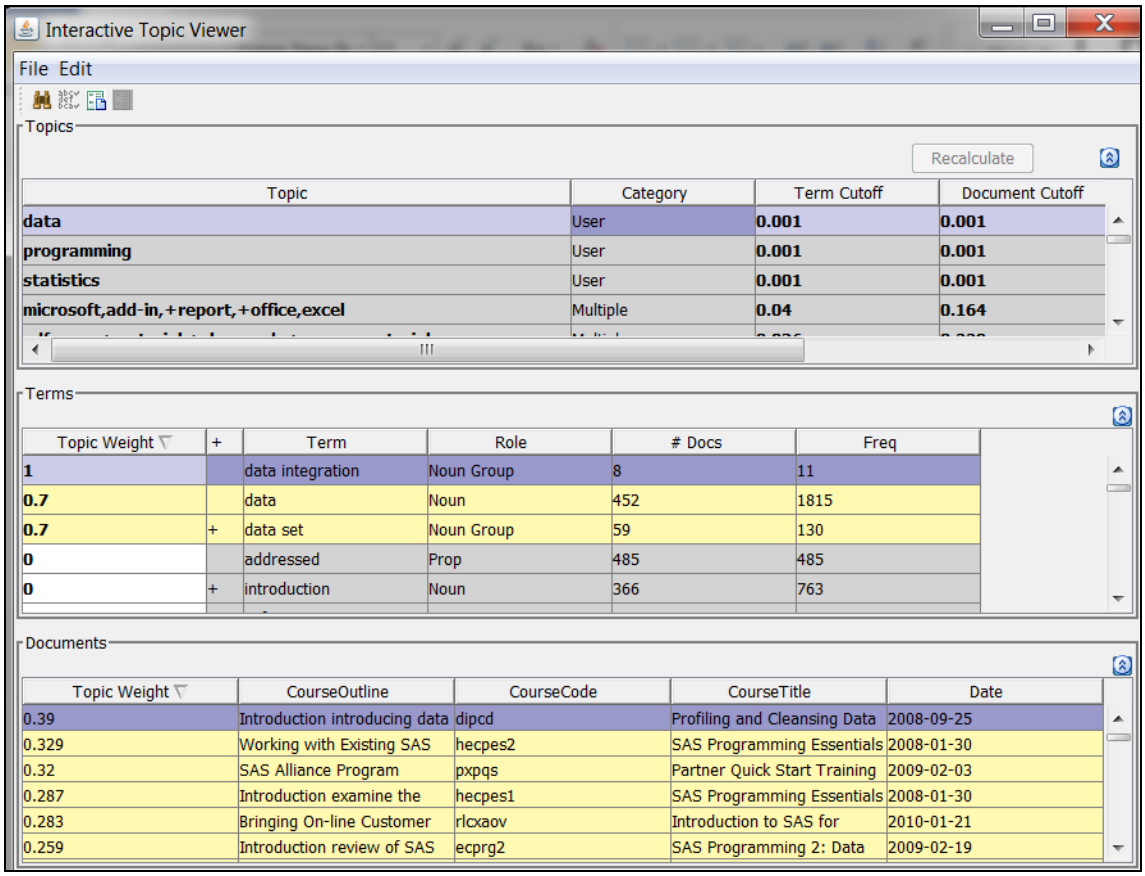
After you attached a **Text Topic** node to the **Text Filter** node, you were asked to go the Property Sheet. Under **User Topics**, open the customized topic list, **DMTXT.SASTOPICS**. After you run the Text Topics node with the other settings retained as defaults, open the Results window.



The three bars to the far left in both bar chart windows relate to the three custom topics: **data**, **programming**, and **statistics**, which were specified in the **DMTXT.SASTOPICS** data set. The remaining bars relate to (automatically) derived topics. The Number of Terms by Topics bar chart reveals that only a few terms were used to define the custom topics. Perhaps more terms should be used. The Number of Documents by Topics bar chart reveals that the custom topics are more prevalent than the derived topics. The smallest custom topic, *programming*, appears in 253 documents. The most popular automatically derived topic appears in 96 documents. (See the arrow pointing to the bar.) You can get the topic frequencies by positioning the cursor over the bar related to a topic. The plots are dynamic.

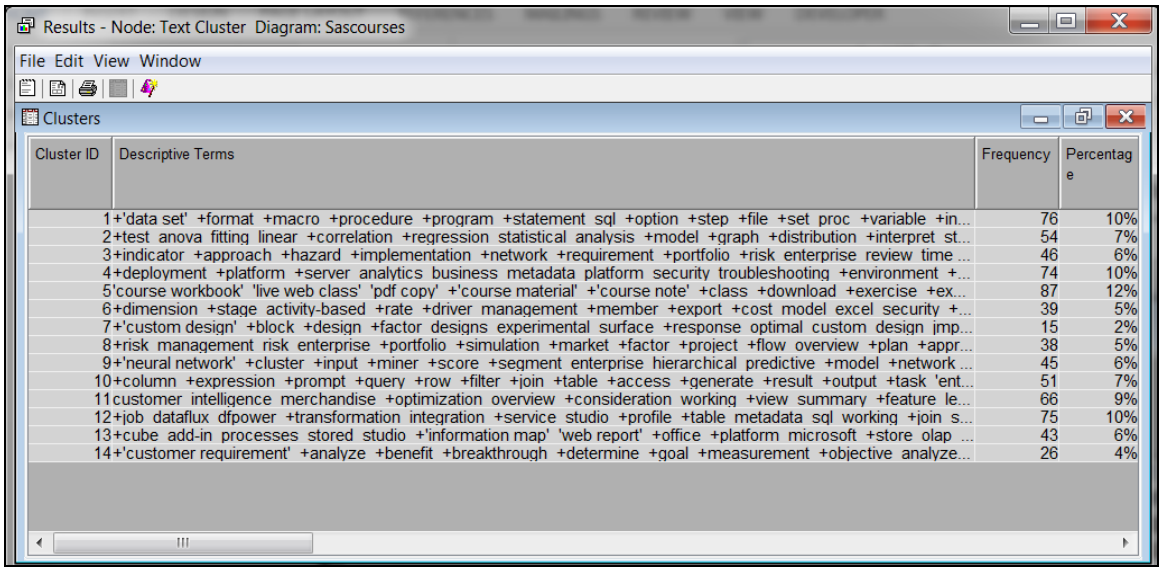
Close the Results window. You were asked to determine what variables were created by the Text Topic node. Opening the Exported Data for this node shows that the variables **TextTopic_1** to **TextTopic_28** and **TextTopic_raw1** to **TextTopic_raw28** were created. Twenty-eight topics were generated (three user topics + 25 derived topics).

Open the Interactive Topic Viewer through the Property Sheet. The following window appears:



A custom topic is similar to a predefined query. The first three topics are always the user-defined ones (in this case: **data**, **programming**, and **statistics**). The topic weight shown in the Documents window determines whether the topic is present. (That is, the query is satisfied.) If the topic weight exceeds the document cutoff, then the document is classified as having the topic. Close the Topic Viewer.

Running the Text Cluster node with default settings leads to a 14-cluster solution. Maximize the cluster table and examine the descriptive terms for each cluster.



The descriptive terms help identify the courses that appear in each cluster. Careful scrutiny reveals that Cluster 1 is a programming cluster and Cluster 2 is an ANOVA and regression cluster. Cluster 7 appears to be a design-of-experiments cluster. It would also be useful to read some of the documents in each cluster to better understand what types of documents belong to a cluster. Because courses often contain material from several subjects, clustering into mutually exclusive categories might be less useful than the topics created from the Text Topic node.

Looking at the Exported Data for the Text Cluster node shows that 36 SVD variables were generated.

Solutions to Student Activities (Polls/Quizzes)

2.03 Multiple Answer Poll – Correct Answers

Which of the following tasks can be performed with the Text Import node?

a. perform optical character recognition (OCR) of embedded bitmaps in document files
b. convert Microsoft Word, Excel, and PowerPoint files to ASCII text
c. process documents having more than 32,000 characters
d. act as a web crawler or robot to fetch and convert Internet pages to ASCII text files

15

2.04 Multiple Choice Poll – Correct Answer

How many of the 11 clusters and the 25 topics were clearly associated with the Unabomber paragraph extracts? (Use the cluster and topic descriptive terms to answer this question.)

a. 1 cluster and 2 topics
b. 2 clusters and 1 topic
c. 2 clusters and 2 topics
d. 1 cluster and 1 topic

25

Chapter 3 Algorithmic and Methodological Considerations in Text Mining

3.1 Methods for Parsing and Quantifying Text

Objectives

- Explain tokenization and describe the transition from tokens to words in a language.
- Define frequency (local) weights and term (global) weights and describe how they are used.
- Provide guidelines for choosing weights.
- Explain the basic vector (metric) space model for representing documents and terms.
- Explain how singular value decomposition projects documents and terms into a smaller dimensional metric space.

3

Text Mining Definitions

Corpus

A collection of documents is called a *corpus*.

Tokens, Separators, and Terms

A document consists of a set of tokens. A *token* is a contiguous string of characters that does not contain a separator. A *separator* is a special character such as a blank or mark of punctuation. A *term* is a token or a sequence of tokens (such as *White House*) with a specific meaning in a given language.

4

Types of Text Extraction Ordered By Increasing Complexity

1. Token extraction
2. Term extraction (token + language ⇒ term)
3. Concept extraction (nouns, noun phrases)
4. Entity extraction (associates nouns with entities – for example, Person: Mr. White, Location: White House)
5. Atomic fact extraction (associates nouns with verbs, that is, subject ⇒ action – for example, terrorist ⇒ bombed)
6. Complex fact extraction (natural language understanding)

(Wakefield 2004)

5

The approach for parsing and quantifying your text data will vary based on the task that you want to perform. In this chapter, we discuss what is available in SAS Text Miner. In other text analytic products, other techniques are used. For example, SAS Sentiment Analysis provides capabilities for atomic fact extraction. However, many atomic fact extraction exercises must be customized. For example, you could train a predictive model to mimic how a domain expert assigns categories. Supervised classification often requires problem-specific tasks related to data preparation and model building.

Characteristics of a Document

A document consists of the following elements:

- letters
- words
- sentences
- paragraphs
- punctuation
- possible structural items (chapters, sections)

The elements of a document can be counted and compared across documents.

6

Various weighting strategies are introduced later to modify simple counts for the terms.

Zipf's Law

Let $t_1,t_2,\ldots,t_n$ be the terms in a document collection arranged in order from most frequent to least frequent.

Let $f_1,f_2,\ldots,f_n$ be the corresponding frequencies of the terms. The frequency f_k for term t_k is proportional to $1/k$.

Zipf's law and its variants help quantify the importance of terms in a document collection. (Konchady 2006)

"The product of the frequency of words (f) and their rank (r) is approximately constant."

7

In practice, Zipf's Law is derived as a Power Law, with free parameters that can be estimated based on the document collection. The general formula is shown here:

$$f_k = C/(\omega + k)^{\theta}$$

where C is a constant such that, for given ω and θ, $\sum_{k=1}^{n} f_k = T$, the total number of words in the document collection. The parameters ω and θ are estimated for a given document collection.

Konchady (2006) relates Zipf's Law to quantifying the importance of a term: "…the number of meanings of a word is inversely proportional to its rank." (Konchady 2006, page 87)

Application of Zipf's Law permits identification of important terms for purposes such as describing concepts or topics. You will not encounter Zipf's Law (or similar theoretical laws) directly, but you can see the results of Zipf's Law in text mining applications (for example, in the list of terms used to define a topic). Along with methods such as Hidden Markov Models (HMM), the implementation is often hidden from the user. Only the results of the methodology are visible.

Relevance of Zipf's Law to Text Mining

- Often, a few, very frequent terms are not good discriminators.
  - *stop* words, for example, the, and, an, or, of
  - often words that are described in linguistics as *closed-class* words, which is a grammatical class that does not get new members
- Typically, there is the following in a document collection:
  - a high number of infrequent terms
  - an average number of average frequency terms
  - a low number of high frequency terms

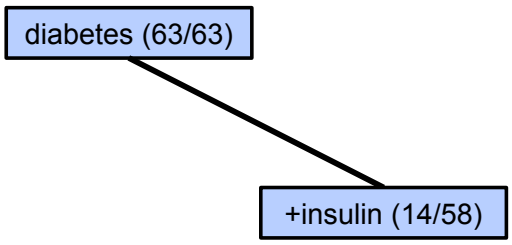
✎ Terms that are neither high nor low frequency are the most informative.

8

Frequency filtering is suggested by Zipf's Law.

Conditional Counts: Concept Linking

Centered term: a term that is chosen to investigate



Concept linked term: a term that co-occurs with a centered term

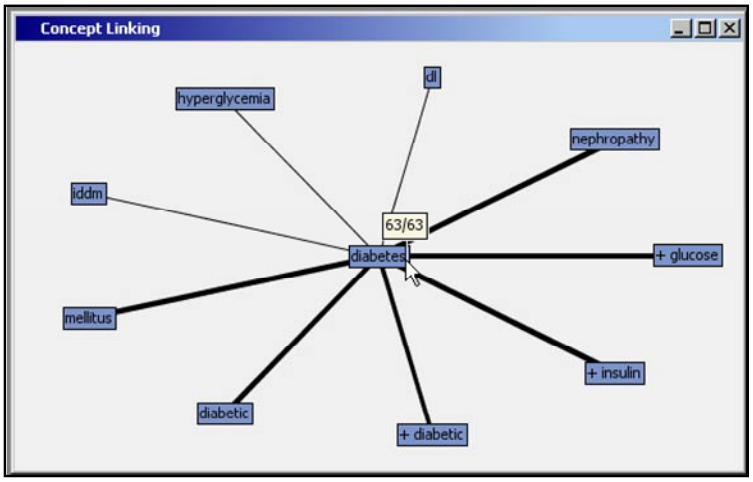
✎ In this diagram, the centered term is *diabetes*, which occurred in 63 documents. The term *insulin* (and its stemmed variations) occurred in 58 documents, 14 of which also contained *diabetes*.

9

continued...

Concept linking is available in the Interactive Filter Viewer of the Text Filter node. In the viewer, access the Terms table, select a term, right-click, and select **View Concept Links**.

Conditional Counts: Concept Linking



The term *diabetes* occurs in 63 documents.

10

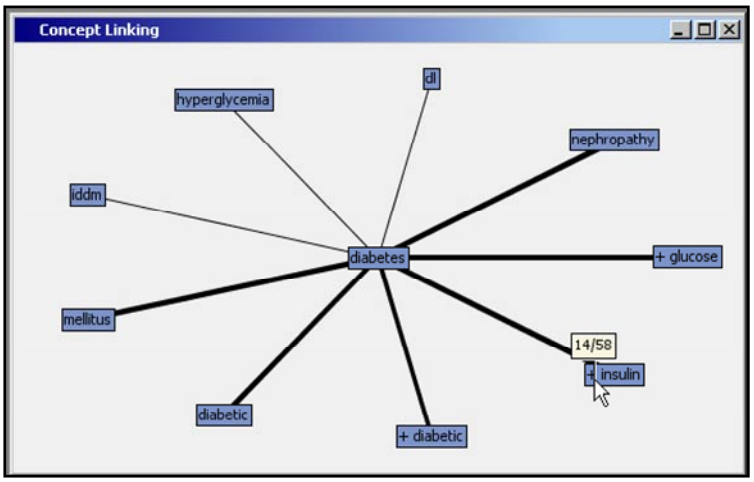
continued...

The Reference Help provides the following description:

"The width of the line between the centered term and a concept link represents how closely the terms are associated. A thicker line indicates a closer association."

The actual metric used to judge association strength is not given.

Conditional Counts: Concept Linking

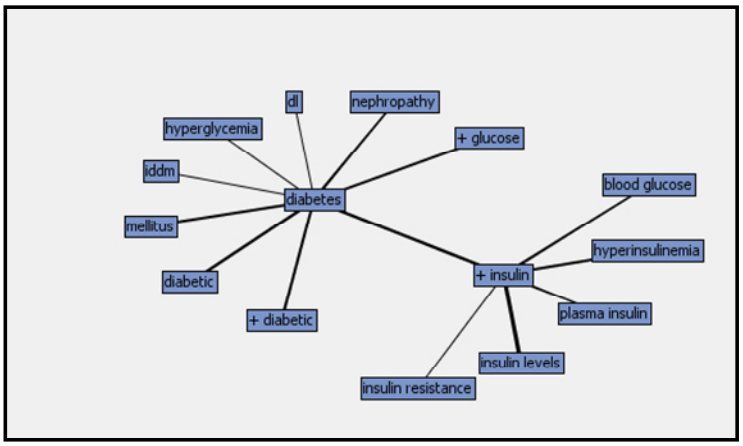


The term *insulin* and its variants occur in 58 documents, and 14 of those documents also contain the term *diabetes*.

11

continued...

Conditional Counts: Concept Linking



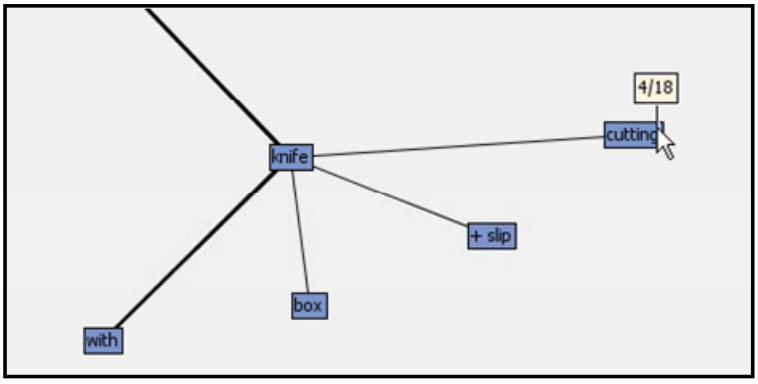
Terms that are primary associates of *insulin* are secondary associates of *diabetes*.

12

Although the concepts related to document and term distances, associations, and similarities are relevant for text mining, the raw frequency counts of terms in documents are typically too primitive to be used for text mining. Weighting strategies and sophisticated linear algebra techniques help move from counting words to extracting concepts.

Setup for the Poll

Consider the following section of a concept linking plot:



13

3.01 Multiple Choice Poll

How many documents contain the term *cutting* as indicated in the setup slide?

a. 18
b. 4
c. 14
d. 22

14

Quantification Steps

The basic strategy for the quantification of free form text with the Text Miner nodes involves the following:

- obtaining the corpus of terms that will be used after applying stemming, synonym creation, filtering, and so on
- representing each document and each term in a vector space via the document by term (or term by document) matrix
- projecting the documents and terms into a lower dimensional vector space
- conducting clustering and topic generation for the documents in this lower dimensional vector space

16

Raw Document by Term Matrix

- The raw document by term matrix shows the frequencies that each term was used in each document. Here you can think of documents as observations and terms as variables.

		(Var 1) apple	(Var 2) cat	(Var 3) cats	(Var 4) dog	(Var 5) dogs	(Var 6) farm	... ...	(Var 4999) White House	(Var 5000) Senate
(Obs 1)	Doc 1	1	1	2	2	0	1	...	0	0
(Obs 2)	Doc 2	0	1	0	1	1	0	...	3	2
(Obs 3)	Doc 3	0	1	0	0	1	0	...	4	4
(Obs ...)	...	...	...	...	...	...	...	...	...	...
(Obs N)	Doc N	2	2	2	3	0	1	...	0	0

Document by Term Matrix

- For the table in this slide, each document is represented by a row vector of 5000 frequencies.
- Doc 1 has the row vector (1, 1, 2, 2, 0, 1, …, 0, 0).
- Notice that Doc 1 and Doc N have somewhat similar vector values, as do Doc 2 and Doc 3.

17

Obtaining document by term frequencies shows how documents can be represented in a vector space whose elements are the frequencies of each term. However, this is likely to be a high-dimensional space with many 0 values. The dimensionality can be reduced by language processing steps such as stemming, synonym creation, and filtering out low frequency terms.

Applying Stemming, Filtering, and So On

- In the previous document by term table, you saw that each document was represented by 5000 terms. That is quite a lot of variables.
- By stemming terms, such as putting *cat* and *cats* together, you reduce the number of columns of the document by term matrix.
- Applying synonyms and filtering out very common and very rare terms also reduce the number of columns. In this example, you go from 5000 to 1000 terms.

		(Var 1) apple	(Var 2) cat (stemmed)	(Var 3) dog(stemmed)	(Var 4) farm	... ...	(Var 999) White House	(Var 1000) Senate
(Obs 1)	Doc 1	1	3	2	1	...	0	0
(Obs 2)	Doc 2	0	1	2	0	...	3	2
(Obs 3)	Doc 3	0	1	1	0	...	4	4
(Obs ...)	...	...	...	...	...	...	...	...
(Obs N)	Doc N	2	4	3	1	...	0	0

Reduced Document by Term Matrix after Stemming, Filtering, Synonyms, and so on

18

The basic data table of document by term frequencies can of course be transposed into the term by document frequency matrix.

Transposing: Term by Document Matrix

- Transposing the table into a term by document matrix of course provides exactly the same information.
- This term by document matrix is often the one presented for analytic purposes.
- One can also think of the terms as the objects and the documents as the variables. The term *apple* is represented by the vector (1,0,0,…,2).
- In this table, the terms *White House* and *Senate* have similar row vectors.

		(Var 1) Doc 1	(Var 2) Doc 2	(Var 3) Doc 3	(Var …) …	(Var N) Doc N
(Obs 1)	apple	1	0	0	…	2
(Obs 2)	cat (stemmed)	3	1	1	…	4
(Obs 3)	dog(stemmed)	2	2	1	…	3
(Obs 4)	farm	1	0	0	…	1
…	…	…	…	…	…	…
(Obs 999)	White House	0	3	4	…	0
(Obs 1000)	Senate	0	2	4	…	0

Transposing: Term by Document Matrix after Stemming, Synonyms, and so on

19

However, there are various problems with this type of data. Even after stemming and filtering, there are often a large number of terms remaining, so there is still the difficulty of a high-dimensional vector space. Also, the data matrix is very sparse. There are usually 90% of the document-term frequencies that are 0. Furthermore, by Zipf's law, the frequency counts of terms are very long tailed. That is, there is a small number of very common terms that are used over and over again in most of the documents.

The Sparse, High-Dimensional Vector Spaces

- After the frequency counts are obtained, you see that both terms and documents can be represented in vector spaces.
- However, in both cases, even after stemming and other filtering steps have been applied by the Text Parsing and Text Filter nodes, you usually still face a very high-dimensional data set.
- In addition, the matrices of frequency counts are very sparse because many words appear only in just 1 or 2 documents. Typically, 90% or more of the cells in the matrices are 0.
- Also, the frequency counts are highly skewed, as shown by Zipf's law. A small number of words occur many times.

20

The dimensionality and sparseness problems will be addressed by projecting the document and term vector spaces into a lower dimensional space by means of a key theorem from linear algebra referred to as the *singular value decomposition* (*SVD*). Before applying SVD, however, it has been found that weighting the raw document-term cell counts usually produces better text mining results. Weighting also helps alleviate the problem of the skewness of the higher frequency terms by making them less influential.

Handling These Problems

- The problems of high dimensionality and sparseness will be addressed by the application of a key theorem in linear algebra called *singular value decomposition* (*SVD*), as discussed shortly.
- The problem of skewed frequency counts is addressed by applying weights to the frequencies.
- This is a two-tiered weighting scheme controlled in the Text Filter node:
  - *Local weights* L_{ij}, also called *frequency weights*, are *calculated for term* i *in document* j.
  - Term weights G_i, also called global weights, are calculated for term i.
  - The final weight for each cell is the product $G_i L_{ij}$.

21

Frequency weights, which are often called *local weights* in the text mining and information retrieval literature, are the first step in transforming the raw cell counts. (Actually, frequency weights are a function of the raw cell counts, and the following three functions can be chosen by the user.)

Step 1: Frequency Weights (Local Weights)

- There are three options for the frequency weights in the Text Filter node:

None	$L_{ij} = a_{ij}$
Binary	$L_{ij} = \begin{cases} 1 \text{ if term } i \text{ is in document } j \\ 0 \text{ otherwise} \end{cases}$
Log	$L_{ij} = \log_2(a_{ij} + 1)$

  a_{ij} is the number of times that term i appears in document j.
- The default is Log.

22

Weighted Term-Document Frequency Matrix

Documents

Term	ID	D1	D2	⋯	D*n*
T1	1	$L_{1,1}$	$L_{1,2}$	⋯	$L_{1,n}$
T2	2	$L_{2,1}$	$L_{2,2}$	⋯	$L_{2,n}$
⋮					⋮

L_{ij} = frequency weight for term i and document j

23

Term weights, often called *global weights* in the literature, modify frequency weights to adjust for document size and term distribution.

Step 2: Term Weights (Global Weights)

- There are four options for choosing the term weights for term G_i.
  - Entropy (default when no target present)
  - Inverse Document Frequency (IDF)
  - Mutual Information (only used with a target and the default when a target is present)
  - None

24

A brief discussion of the formulas behind the weights begins below. Although you might gain some insight by looking at the mathematics, experimentation rather than intuition is often the best strategy for choosing weights. Experience with similar text analytic problems can help you develop your own guidelines.

Term Weight Formulas

$a_{i,j}$ = frequency that term i appears in document j

g_i = frequency that term i appears in document collection

n = number of documents in the collection

d_i = number of documents in which term i appears

$p_{i,j} = a_{i,j} / g_i$

25

continued...

Term Weight Formulas

Entropy

$$G_i = 1 + \sum_{j=1}^{d_i} \frac{p_{ij} log_2(p_{ij})}{log_2(n)}$$

$$0 \leq G_i \leq 1$$

Low Information ⟶ High Information

26

continued...

The entropy term weight is slightly misnamed here. The usual definition of entropy from Shannon's (1948) information theory is the expression $-\sum_{j=1}^{d_i} p_{ij} \log_2 p_{ij}$, so a better way to describe the term weight used here would be 1 – normalized entropy..

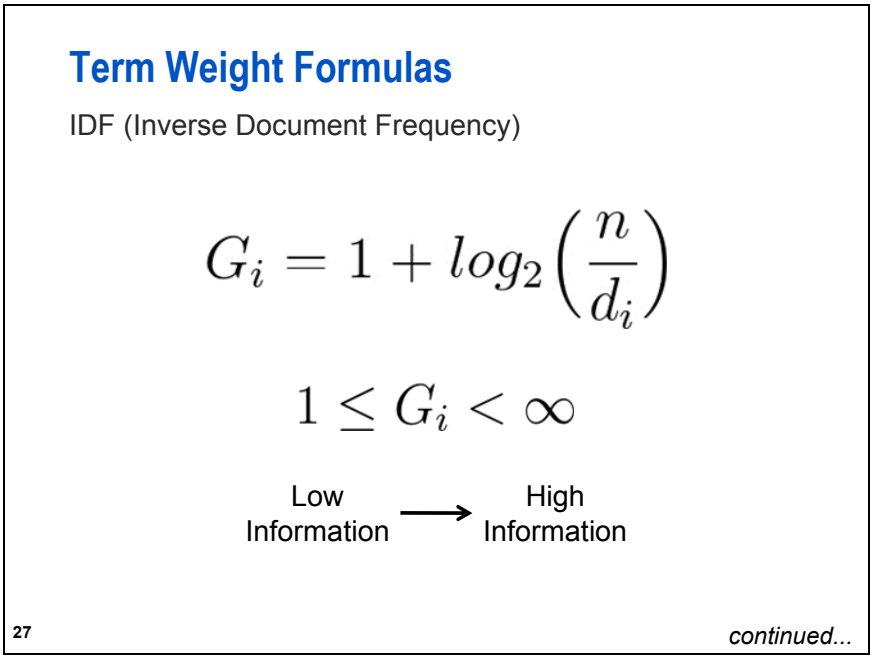
Because the logarithm of zero is undefined, the product in the numerator is taken to be zero if the proportion p_{ij} is zero. Two simple cases illustrating the calculation of this term weight are shown below.

Simple Case 1: Term *i* occurs one time in only one document:

$$G_i = 1 + \frac{(1/1)\log_2(1/1)}{\log_2(n)} = 1 + \frac{(1)(0)}{\log_2(n)} = 1$$

Simple Case 2: Term *i* occurs one time in each of the total *n* documents.

$$G_i = 1 + \frac{\sum(1/n)\log_2(1/n)}{\log_2(n)} = 1 + \frac{n(1/n)(-\log_2(n))}{\log_2(n)} = 1 + \frac{-\log_2(n)}{\log_2(n)} = 1 - 1 = 0$$



If a term appears in every document, then the IDF weight is 1 because then $d_i = n$. The maximum weight for a fixed document collection occurs when the term appears in exactly one document, and the weight becomes $1 + \log_2(n)$. No upper limit exists because the number of documents *n* in a collection can be arbitrarily large.

Entropy and IDF weights achieve a maximum when exactly one term appears in exactly one document. This implies a very discriminating term, but not a very useful one, because it occurs in only one document. In fact, by default, the Text Filter node removes terms that do not occur in at least four documents, although this is under user control. Both weights are at minimum or near minimum if a term appears exactly one time in every document. In this case, the term is not very discriminating because it occurs everywhere throughout the collection of documents.

Term Weight Formulas

Mutual Information

G_i = max(over k) $\left[\log_{10} \left(\frac{P(t_i, C_k)}{P(t_i)P(C_k)} \right) \right]$

where

$C_1, C_2, \ldots, C_k$ are the k levels of a categorical target variable.

$P(t_i)$ is the proportion of documents containing term i.

$P(C_k)$ is the proportion of documents having target level C_k.

$P(t_i, C_k)$ is the proportion of documents where term i is present and the target is C_k.

(Note that $0 \leq G_i < \infty$ and the log is base 10.)

28

Although G_i is theoretically unbounded, in practice it is usually less than 1. Here is a simple example showing how it is calculated for the case of a binary target, where k=2:

	Number of documents with target level C_1	**Number of documents with target level C_2**	
Number of docs where term t_i not present	**100**	**25**	**125**
Number of docs where term t_i is present	**10**	**50**	**60**
	110	**75**	**185**

From this crosstabulation, we get

$$P(t_i) = 60/185 = .324$$
$$P(C_1) = 110/185 = .595$$
$$P(C_2) = 75/185 = .405$$
$$P(C_1, t_i) = 10/185 = .054$$
$$P(C_2, t_i) = 50/185 = .270$$
$$G_i = \max \left\{ \log_{10} \left(\frac{.054}{.324 * .595} \right), \log_{10} = \left(\frac{.270}{.324 * .405} \right) \right\} = .313$$

Generalizing this to the case of a categorical target with $k>2$ merely requires extending the crosstabulation to a 2 by k table and then computing the individual factors in the same way as above.

Multiplying the local and global weights produces an adjusted count that is often superior to using raw counts alone.

Weighted Term-Document Frequency Matrix

After the frequency (local) and term (global) weights have been calculated for each term, the final weights used are the product of the two.

$\hat{a}_{i,j} = G_i L_{i,j}$

G_i is the term (global) weight for term i.

$L_{i,j}$ is the frequency (local) weight for term i in document j.

29

The term-document frequency matrix, weighted or unweighted, is the foundation of the linear algebra approach to text mining.

Weighted Term-Document Frequency Matrix

Terms	Documents: D1	D2	Dn
T1	$\hat{a}_{1,1}$	$\hat{a}_{1,2}$	$\hat{a}_{1,n}$
T2	$\hat{a}_{2,1}$	$\hat{a}_{2,2}$	$\hat{a}_{2,n}$
Tm	$\hat{a}_{m,1}$	$\hat{a}_{m,2}$	$\hat{a}_{m,n}$

30

Term Weight Guidelines

- When a target is present, Mutual Information is the default. It is a good choice when it can be used.
- Entropy and IDF weights give higher weights to rare or low frequency terms.
- Entropy and IDF weights give moderate to high weights for terms that appear with moderate to high frequency, but in a small number of documents.
- Entropy and IDF weights vary inversely to the number of documents in which a term appears.
- Entropy is often superior for distinguishing between small documents that contain only a few sentences.
- Entropy is the only term weight that depends on the distribution of terms across documents.

31

continued...

Term Weight Guidelines

- Remember, you can suppress both frequency weights and term weights by choosing the option None for each of these in the Text Filter node.
  - If you choose None, then the raw cell counts will be analyzed down stream, that is,, $\hat{a}_{i,j} = a_{i,j}$.
- Be experimental. Try different weight settings to find what gives you the most interpretable or most predictive results for your data.

32

A simulation study artificially creates a document collection and distributes terms across the documents using various strategies (for example, creating rare terms and creating terms with frequency counts that follow a certain distribution). Even though the data set is completely artificial and simple, it is informative to examine these results.

Simulation Study of Term Weights

Term	Term Freq	Doc Freq	Entropy	IDF	Mutual Information
armadillo	102	2	0.8495	6.6439	0.4943
bear	105	64	0.1264	1.6439	0.1839
cat	113	59	0.1405	1.7612	0.0421
cow	110	66	0.1107	1.5995	0.2177
dog	107	66	0.1183	1.5995	0.0478
gopher	106	55	0.1580	1.8625	0.2665
hamster	109	65	0.1194	1.6215	0.4308
horse	109	62	0.1315	1.6897	0.1818
kitten	105	62	0.1303	1.6897	0.0307
moose	1934	100	0.0973	1.0000	0.0000
mouse	108	63	0.1296	1.6666	0.0943
otter	1	1	1.0000	7.6439	0.4943
pig	107	58	0.1440	1.7859	0.0592
puppy	115	58	0.1576	1.7859	0.5447
raccoon	967	50	0.2478	2.0000	0.1086
seal	10	10	0.5000	4.3219	0.2712
squirrel	100	100	0.0000	1.0000	0.0000
tiger	117	70	0.1027	1.5146	0.0070
walrus	25	25	0.3010	3.0000	0.0480
zebra	3812	100	0.1008	1.0000	0.0000

33

You can verify the IDF calculations using the Doc Freq column and noting that there are 100 documents in the simulation. For example, for the term *armadillo*, the IDF term weight is as follows:

$$1 + \log_2(100 / 2) = 1 + \log_2(50) = 6.6439$$

The gray scale version is difficult to interpret, but the color version of the table highlights low information terms with a red background, and high information terms with a green background.

The two terms in the mutual information column are the two terms that defined the target variable. If a document contained both the term *hamster* and the term *puppy*, then the target variable was assigned a value of 1. Otherwise, it received a value of 0. Two terms received a higher target-based term weight than *hamster* because the two terms by happenstance appear less frequently than *hamster* when the term *puppy* is not present. In particular, by happenstance, the one document that contains *otter* also contains *hamster* and *puppy*, and the two documents that contain *armadillo* also contain both *hamster* and *puppy*.

The results show that entropy and IDF weights tend to produce similar results. IDF is recommended for larger documents, whereas entropy might be more appropriate for smaller documents. Of course, these results cannot be extrapolated to all document collections. In particular, a typical document in the simulated collection is small, so the results would be more useful for document collections such as the Medline data, but less useful for multi-page reports.

Note also that in the Mutual Information column, the term *puppy* has the highest weight, and the term *hamster* has the fourth highest weight. Because these two terms were used to define the target variable, it obviously makes good sense that these would have high weights because Mutual Information takes the target variable into account. The other terms that have very high Mutual Information weights, *armadillo* and *otter*, have spuriously high values, which are due to the tiny number of documents that these occurred in (two for *armadillo* and one for *otter*). In fact, by default, the Text Filter node drops terms that occur in fewer than four documents, so these terms would not have been kept unless this default value was changed.

3.02 Multiple Choice Poll

Which term weight is recommended and is the default for documents that are associated with a categorical target variable?

a. IDF
b. mutual information
c. entropy
d. none (G=1)

34

3.2 Dimension Reduction with SVD

Objectives

- Sketch how singular value decomposition (SVD) is used to project the high dimensional document and term spaces into a lower dimension space.
- Illustrate what is happening with a simple example.
- Discuss Text Topic and Text Cluster results in light of the SVD.

37

3.03 Multiple Choice Poll

Which response best describes your preference?

a. I am eager to learn the technical details of singular value decomposition.
b. I would like to understand the concepts that are important, but I would prefer to skip the math.
c. I do not really care how SVD works. I only want to know how to use the software to solve my problem.
d. I am only here to keep from doing real work.

38

Linear Algebra and SVD in Text Mining

- You have seen how the main data set in the analysis of free-form text consists of a term-document matrix.
- You now assume at this point that all the natural language parsing and tokenization of terms, the application of start or stop lists, filtering, weighting, and so on, have been performed so that you can focus on the final version of the term-document matrix.
- Linear algebra includes the study of matrices and matrix properties.
- Professor Gilbert Strang of MIT, a world expert on this topic, has referred to SVD as “The Fundamental Theorem of Linear Algebra.”

39

Statement of SVD Theorem

- This brief discussion will be based on the very helpful paper "Taming Text with the SVD" (recommended reading and readily available for downloading from the Internet) by Dr. Russ Albright of SAS R&D.
- The most relevant aspects of the SVD theorem are presented for the purpose of dimension reduction, followed by an example.
- If you do not understand the abstract explanations, the concrete example will at least give you the "flavor" of what is happening. (Also see the optional exercise at the end of Section 3.2.)
- Define A to be a term-document matrix with m terms and n documents. (Typically, $m > n$. That is, there are more terms than documents.)

40

Statement of SVD Theorem

- The SVD theorem states that the term-document matrix (and, in fact, ***any*** rectangular matrix of real or complex values) can always be decomposed into the product of three matrices in the form $A = U\Sigma V^T$:

  T signifies the transpose of a matrix.

  r is the rank of the matrix A.

  U is an m x r matrix satisfying the orthogonality condition $U^TU=I_{r \times r}$.

  $I_{r \times r}$ is an r x r identity matrix.

  Σ is an r x r diagonal matrix consisting of r positive "singular values"
  $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$

  V is an r x n matrix satisfying the orthogonality condition $VV^T=I_{r \times r}$.
- The singular values σ_i can be thought of as providing a measure of importance used to decide how many dimensions to keep.

41

SVD Example

- Let's look at Russ Albright's little example consisting of three documents:
  - Doc 1: Error: invalid message file format
  - Doc 2: Error: unable to open message file using message path
  - Doc 3: Error: Unable to format variable
- These three documents generate the following 11 x 3 term-document matrix A.

		doc 1	doc 2	doc 3
Term 1	error	1	1	1
Term 2	invalid	1	0	0
Term 3	message	1	2	0
Term 4	file	1	1	0
Term 5	format	1	0	1
Term 6	unable	0	1	1
Term 7	to	0	1	1
Term 8	open	0	1	0
Term 9	using	0	1	0
Term 10	path	0	1	0
Term 11	variable	0	0	1

42

continued...

SVD Example

- With the right software (for example, PROC IML), it is very easy to compute the SVD decomposition for this little example and obtain the separate matrices U, Σ, and V.
- The product $U^T A$ produces the SVD projections of the original document vectors. These are the document SVD input values that you have seen, which are produced by the Text Cluster variable (except that they are normalized for each document as explained on a later slide).
- This amounts to forming linear combinations of the original (possibly weighted) term frequencies for each document.

43

continued...

SVD Example

- First project the first document vector d_1 into a three-dimensional SVD space by the matrix multiplication:

$$U^T d_1 = \begin{bmatrix} 0.43 & 0.11 & 0.55 & 0.33 & 0.21 & 0.31 & 0.31 & 0.22 & 0.22 & 0.22 & 0.09 \\ 0.30 & 0.13 & -0.37 & -0.12 & 0.55 & 0.18 & 0.18 & -0.25 & -0.25 & -0.25 & 0.43 \\ 0.11 & 0.52 & 0.2 & 0.36 & 0.27 & -0.41 & -0.41 & -0.16 & -0.16 & -0.16 & -0.25 \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

U^T was obtained using the SVD matrix function in PROC IML applied to the A matrix

d_1 = term-frequency vector for document 1 (using the unweighted counts here)

- The product of the 3 x 11 U^T matrix with the 11 x 1 term-frequency vector d_1 for doc 1 gives:

$$U^T d_1 = \hat{d}_1 = \begin{bmatrix} 1.63 \\ 0.49 \\ 1.45 \end{bmatrix}$$

- And then write this in transposed form with column labels:

$$\hat{d}_1^T = \begin{bmatrix} \text{SVD1} & \text{SVD2} & \text{SVD3} \\ 1.63 & 0.49 & 1.45 \end{bmatrix}$$

44

continued...

SVD Example

- The SVD dimensions are ordered by the size of their singular values ("their importance"). Therefore, the document vector can simply be truncated to obtain a lower dimensional projection:
  - 2-D representation for doc 1 is $\begin{bmatrix} \text{SVD1} & \text{SVD2} \\ 1.63 & 0.49 \end{bmatrix}$
- As a final step, the Text Cluster node then normalizes the coordinate values so that the sums of squares for each document are 1.0:
  - Using this document's 2-D representation, $1.63^2 + .49^2 = 2.847$ and $\sqrt{2.897} = 1.70$.
  - Therefore, the final 2-D representation for doc 1 would be $\begin{bmatrix} \text{SVD1} & \text{SVD2} \\ 0.96 & 0.29 \end{bmatrix}$.
- These are the SVD1 and SVD2 values that you would see for this document by looking at the Exported Data coming out of the Text Cluster node.

45

Dimensionality Reduction

- The tiny example given here has a term-document matrix of rank $r=3$. (The rank is always less than or equal to the minimum of the number of documents and the number of terms.)
- In actual practice, the rank of the term-document matrix will usually be in the thousands, so the SVD algorithm is used to dramatically reduce the dimensionality of the data.
- The SVD algorithm derives SVD dimensions in order of "importance" (based on the singular values σ_i).
- The number of SVD dimensions to keep is based on looking at these singular values and establishing a cut-off value k.

46

continued...

Dimensionality Reduction

- The user specifies a maximum dimension *M* (default=100 and highest allowed value=500) for the number of SVD dimensions to keep.
- The SVD algorithm produces the *M* singular values in decreasing order.
- The sum of the *M* singular values (squared) acts as a metric for the amount of information in the document collection. Treating the sum of the top *M* squared values as the "total information" is useful for arriving at a reasonable cutoff.

47

continued...

Dimensionality Reduction

- The user also specifies an SVD Resolution value:
  - High=100%
  - Medium=5/6=83.3%
  - Low=2/3=66.67% (the default)
- Based on these two settings, the Text Cluster node uses a simple algorithm to decide on the final number of SVD dimensions to use.

48

continued...

Dimensionality Reduction

- To illustrate the logic for deciding the number of dimensions to use:
  - Suppose the user sets Max SVD Dimensions=100 and SVD Resolution=Low (66.7%).
  - Assume that you are working with a big document collection so that the rank of the term-document matrix is much larger than 100.
  - Let the sum of the first 100 squared singular values be given as $\sum_{i=1}^{100} \sigma^2_i = C$.
  - The algorithm determines the minimum dimension $k \leq 100$ such that $\sum_{i=1}^{k} \sigma^2_i / C \geq .667$.
  - In the end, k SVD dimensions will be kept.

49

The problem of dimensionality reduction is challenging in a text mining setting. The default settings in the SAS Text Cluster often work very well, but you should be prepared to experiment with the maximum number of SVD dimensions kept, as with many other parameter settings.

Experimenting with the SVD Dimensions

This demonstration illustrates some experimentation with the SVD dimensions.

The ASRS data set can be accessed from the following link:

http://asrs.arc.nasa.gov/

From the website:

"ASRS captures confidential reports, analyzes the resulting aviation safety data, and disseminates vital information to the aviation community."

"More than 850,000 reports have been submitted (through October 2009) and no reporter's identity has ever been breached by the ASRS. ASRS de-identifies reports before entering them into the incident database. All personal and organizational names are removed. Dates, times, and related information, which could be used to infer an identity, are either generalized or eliminated."

As with other data sets used in this course, data sets derived from ASRS have been modified. The original data for this demonstration was extracted from the ASRS, pre-processed, and provided to competitors in a text mining competition sponsored by SIAM and the NASA Ames Research Center. The competition results were presented at the Seventh SIAM International Conference on Data Mining held in 2007 in Minneapolis, Minnesota. Participants were prohibited from using the R language, SAS software, and most commercial software. A link that provides access to the original data follows.

https://c3.nasa.gov/dashlink/resources/138/

A single report in the ASRS database can be a composite derivation of two or more reports filed for the same incident. For example, one runway incursion incident can result in three reports: one from the pilot, one from the copilot, and one from an air traffic controller. An incident involving two or more aircraft can have reports filed from pilots of all aircraft involved, as well as from air traffic controllers. In both examples, there will be only one ASRS report, but that report will be prepared by NASA professionals based on all reports submitted.

Reports can be submitted by aviation professionals, such as pilots, flight attendants, and mechanics. Reports can also be submitted by non-professionals, such as private pilots.

A report in the ASRS database has many fields, with one field representing a primary narrative describing the incident. This primary narrative is stored in the **Text** variable. All of the other fields have been omitted to simplify the text mining component of the analysis. In practice, an automated labeling system would attempt to use all fields.

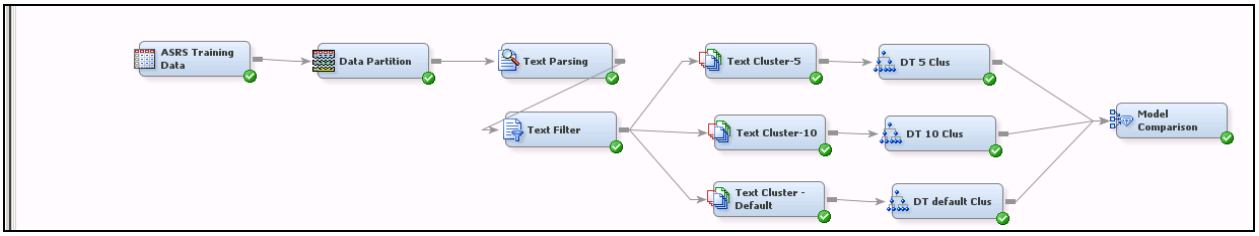
NASA manually assigns to each report 1 or more of 54 anomalies, 1 or more of 32 results, 1 or more of 16 contributing factors, and 1 or more of 17 primary problems. For example, the report might describe an event that was a "runway ground incursion" anomaly, with a "took evasive action" result, that was a "human factor" contributing factor, and a "human factor" primary problem. These fields are not available in the contest data. Instead, the contest data has 22 labels, with a value of 1 "if document i has label j;"

Otherwise, the label has a value of -1. Labels correspond to the topics identified by NASA to aid in the analysis of the reports. The labels are not defined in the competition. For the course data, the 22 labels are named **Target01**, **Target02** up through **Target22**, and an original coding of (-1,1) has been changed to (0,1), with a code of 1, which indicates the presence of the label in the document. A document can be associated with one or more labels.

The ASRS training data we will use contains columns that indicate which of the 22 manually assigned labels relates to a given report. The goal is to develop a system to automatically detect incidents to avoid the time, cost, and error associated with manually labeling the reports. In other words, we will be building a model on a data set where experts have already read the reports and made evaluations. (This is the sort of process that many people will use for sentiment analysis: create a data set of labeled cases and then build an automatic classification/prediction system based on these known cases.) In an actual operational system for this example, we would build 22 models to evaluate whether each of these 22 types of events occurred. This collection of models would provide 22 predicted values, one for each target.

Running the diagram setup for this demonstration will take several minutes.

1. Create a new diagram and name it **SVD_Dimensions**. (As we have stated throughout, in virtually every case this diagram should have been already created for you and the relevant nodes run.) The diagram will look like this when completed:



2. Create a data source for the ASRS training data using **DMTXT.ASRS_TRAINING**. Use the following metadata:

Variables - Ids

(none) not Equal to ...

Columns: Label Mining

Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
ID	**ID**	**Nominal**	**No**		**No**	.	.
Size	**Rejected**	**Interval**	**No**		**No**	.	.
Target01	**Rejected**	**Binary**	**No**		**No**	.	.
Target02	**Target**	**Binary**	**No**		**No**	.	.
Target03	**Rejected**	**Binary**	**No**		**No**	.	.
Target04	**Rejected**	**Binary**	**No**		**No**	.	.
Target05	**Rejected**	**Binary**	**No**		**No**	.	.
Target06	**Rejected**	**Binary**	**No**		**No**	.	.
Target07	**Rejected**	**Binary**	**No**		**No**	.	.
Target08	**Rejected**	**Binary**	**No**		**No**	.	.
Target09	**Rejected**	**Binary**	**No**		**No**	.	.
Target10	**Rejected**	**Binary**	**No**		**No**	.	.
Target11	**Rejected**	**Binary**	**No**		**No**	.	.
Target12	**Rejected**	**Binary**	**No**		**No**	.	.
Target13	**Rejected**	**Binary**	**No**		**No**	.	.
Target14	**Rejected**	**Binary**	**No**		**No**	.	.
Target15	**Rejected**	**Binary**	**No**		**No**	.	.
Target16	**Rejected**	**Binary**	**No**		**No**	.	.
Target17	**Rejected**	**Binary**	**No**		**No**	.	.
Target18	**Rejected**	**Binary**	**No**		**No**	.	.
Target19	**Rejected**	**Binary**	**No**		**No**	.	.
Target20	**Rejected**	**Binary**	**No**		**No**	.	.
Target21	**Rejected**	**Binary**	**No**		**No**	.	.
Target22	**Rejected**	**Binary**	**No**		**No**	.	.
Text	**Text**	**Nominal**	**No**		**No**	.	.

The data set contains 22 variables with the names **Target01** through**Target22**. We will use **Target02** for this demonstration, so all the others should be rejected. From a table in the E.G. Allan et al article "Anomaly Detection Using Nonnegative Matrix Factorization" (2008, p. 215), the incident for **Target02** has to do with noncompliance with policy procedures. The variable **Size** is just the length of the report in bytes and will not be used. Only the report itself (**Text**) is needed, but **ID** can be left as an ID variable.)

3. Drag the **ASRS** data source onto the diagram.

4. To investigate the robustness of the automated assignment, add a **Data Partition** node to the partition data set **DMTXT.ASRS_TRAINING**. Use a 50/50/0 partition.

5. Attach a **Text Parsing** node to the **Data Partition** node. Leave all the defaults as is and run.

6. Attach a **Text Filter** node to the **Text Parsing** node. Change the default weightings to **Log** and **Mutual Information**. (Although in this case, these are the defaults.) Leave all else in default mode and run.

Weightings	
Frequency Weighting	Log
Term Weight	Mutual Information

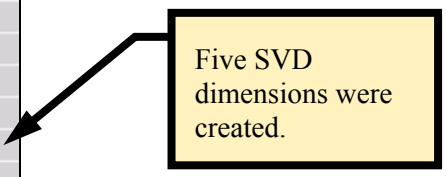
7. Attach a **Text Cluster** node to **Text Parsing** and rename it **Text Cluster – 5.** Change the Transform settings as follows:

Transform	
SVD Resolution	High
Max SVD Dimensions	5

8. This generates a 5-dimensional SVD solution. Run the node and go to **Exported Data** in the **Property Sheet**. Select the **TRAIN** data set and then click the **Explore** button. Verify that you have a 5-dimensional solution by looking at the number of **TextCluster_SVD** variables.

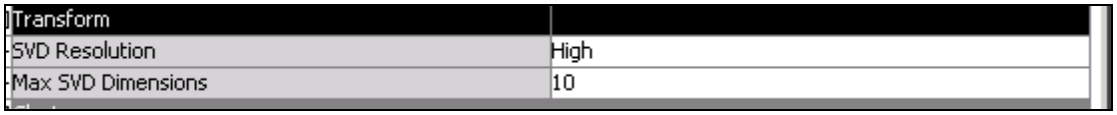
Sample Statistics

Obs #	Variable Name
12	Target09
13	Target10
14	Target11
15	Target12
16	Target13
17	Target14
18	Target15
19	Target16
20	Target17
21	Target18
22	Target19
23	Target20
24	Target21
25	Target22
26	TextCluster_SVD1
27	TextCluster_SVD2
28	TextCluster_SVD3
29	TextCluster_SVD4
30	TextCluster_SVD5
31	TextCluster_cluster_
32	TextCluster_prob1
33	TextCluster_prob2
34	TextCluster_prob3
35	TextCluster_prob4
36	TextCluster_prob5
37	_dataobs_
38	_document_



9. Attach a **Decision Tree** node to the **Text Cluster-5** node. Rename it to **DT 5 Clus**. Change the Assessment Measure property to **Average Square Error** and Leaf Size to **25**. (These are fairly routine changes that are often found to produce better results with trees. Obviously 25 is not some magic number, but the default Leaf Size of 5 is considered by many analysts to be too small.) Run the node, but it is not necessary to look at the results yet.

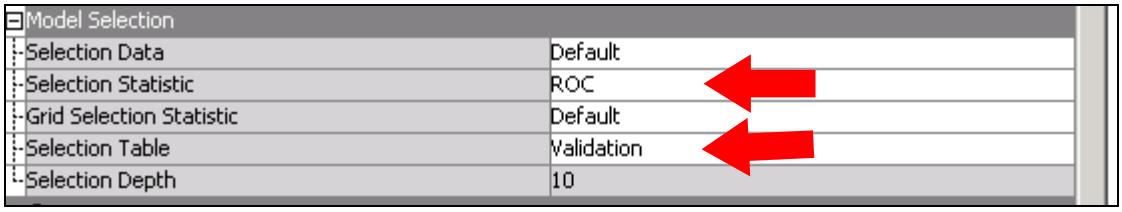
10. Attach another **Text Cluster** node to the **Text Filter** node and rename it **Text Cluster 10**. Change the Transform settings as shown below in order to get a 10-dimensional SVD solution. Run this and verify that you see a set of 10 **TextCluster_SVD** variables.



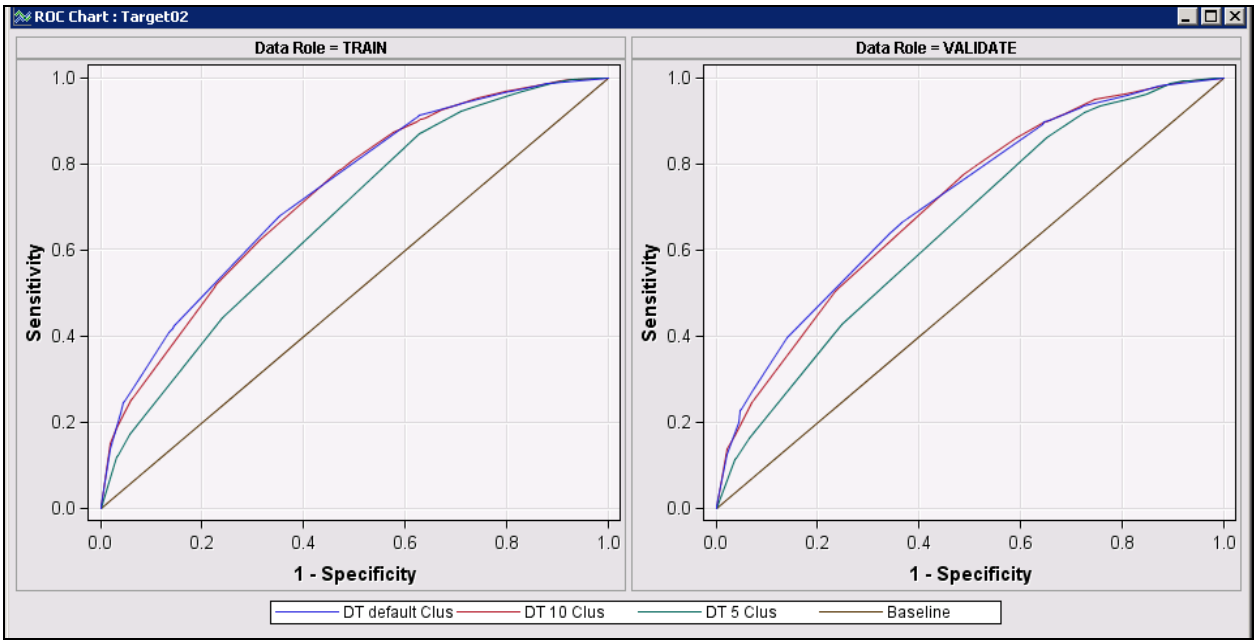
Then copy down your previous **DT 5 Clus** node, rename it **DT 10 Clus**, and connect it to **Text Cluster 10**. Run this tree, but there is no need to look at the results yet.

11. Repeat the previous step with a new **Text Cluster** node renamed as **Text Cluster - Default**. Run this default Text Cluster node and determine that 33 **TextCluster_SVD** variables were produced. Rename this third decision tree to **DT default Clus** and run it.

12. Now connect all three decision trees to the **Model Comparison** node. Set the Property Sheet for **Model Comparison** to be **ROC** for the Selection Statistic property and **Validation** for Selection Table. As a consequence of these changes, the ROC index for the validation data will be shown at the very beginning of the **Model Comparison** results window. Run the **Model Comparison** node and view the results.



13. Open the results and look first at the ROC charts:



In color, it is clear that **DT 5 Clus** is the inferior decision tree model. Examining the Fit Statistics window confirms this: the **DT 5 Clus** tree has a validation ROC index of just .65 compared to the rounded value .71 for the other two models.

Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Roc Index
Y	Tree3	Tree3	DT default Clus	Target02		0.711
	Tree2	Tree2	DT 10 Clus	Target02		0.706
	Tree	Tree	DT 5 Clus	Target02		0.65

To two decimal-place accuracy, **Tree2** and **Tree3** have the same validation ROC index. **Tree2** (**DT 10 Clus**) was generated from a Text Cluster node specifying a 10-dimension solution, whereas **Tree3 (DT default Clus)** was calculated from the default Text Cluster node that generated 33 dimensions.

Remember that the decision tree algorithm itself incorporates variable selection logic. So even though **Tree2** was working with 10 SVD variables as candidate variables generated by the Text Cluster node, it actually only found 8 of them to be useful in the tree. **Tree3** had 33 candidate SVD variables to work with, but only kept 16 for the final tree. (This information can be found by going to tree results and selecting **View** ⇨ **Model** and then viewing the Variable Importance windows.)

The lesson here is that when you have a target variable and are using a modeling method with variable selection logic, the decision about how many SVD variables to keep is less critical. It is better to err on the high side and then let the variable selection algorithm from your model make the choice. Typically, the default settings of **SVD Resolution=Low** and **Max SVD Dimensions=100** will work well in this situation. (You do not want to choose too few dimensions, as we did with **Tree (DT 5 Clus)**, and wind up with an inferior model.)

Exercises

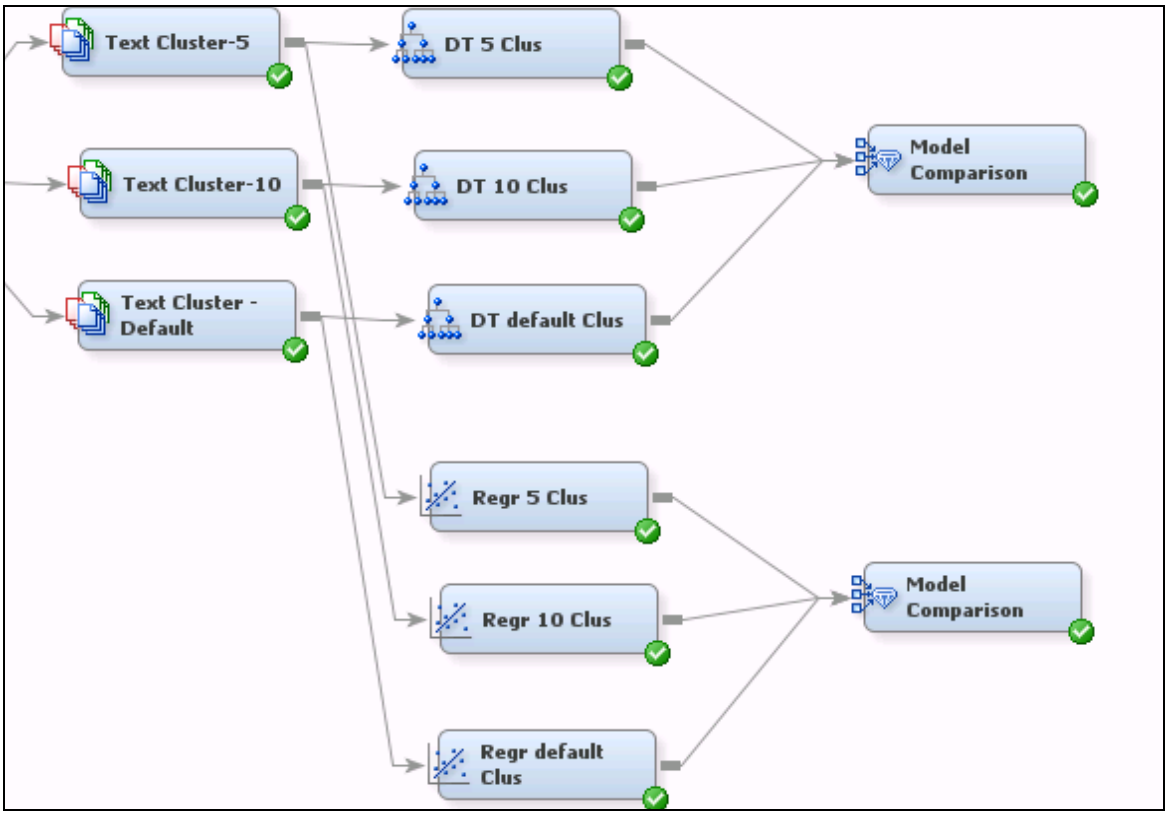
1. Comparing the Effect of the Number of SVD Dimensions Using Regression Models

For this exercise repeat all the steps done in the previous demonstration, but now using three forward selection regression models. The easiest way to do this without having to rerun any of the previous nodes is as follows:

a. Bring down a regression model and connect it to the **Text Cluster-5** cluster node. Rename the regression model to **Regr 5 Clus.** Set it up to do forward selection with assessment on the validation error. Then connect the **Regr 5 Clus** node to a new **Model Comparison** node. (Just copy the previous **Model Comparison** node down.) The reason for using a second Model Comparison node is because with more than three models, the graphs become a little too cluttered to interpret easily.

b. Copy the **Regr 5 Clus** node down and connect it to the **Text Cluster-10** cluster node. Rename this regression model **Regr 10 Clus** and connect it to the second **Model Comparison** node in order to compare it to the first regression model.

c. Copy the first regression node down again. Connect this copy to the **Text Cluster-Default** cluster node. Rename the regression node to **Regr Default Clus**. Again, connect it to the second **Model Comparison** node as with the other two regression models. The ***right part*** of your diagram should now look like this:



d. After running everything through the second Model Comparison node, answer these questions:

1) How many SVD variables were selected by each of the three different regression models?

2) What was the validation ROC index for each of these three models? How do these index values compare with values from the previous three decision tree models?

2. (Optional) Details of the SVD Calculations Performed by the Text Cluster Node

This optional exercise is for those students who are interested in the details of how the document SVD variables are calculated. Although not all users of the SAS Text Analytics software want to know this much, those who do can work through this exercise. In the first step, a very simple text mining project is run to produce the SVD variables computed in the Text Cluster node. In the second step, you will use a PROC IML (Interactive Matrix Language) program to explicitly see how the term-document frequency matrix is analyzed using the SVD algorithm from linear algebra. In the end, you will be able to compare the SVD values from the Text Cluster node to those computed by the PROC IML code and see that they are the same.

You are not expected to write the PROC IML program that is used. It is supplied for you. If you are familiar with matrix algebra and can follow programming logic, you maybe be able to understand the major parts of the internally documented PROC IML program.

a. Create a diagram named **Optional Exercise SVD Calculations**. (This diagram has already been created for you, but you are encouraged to set it up on your own as you have been doing in class.)

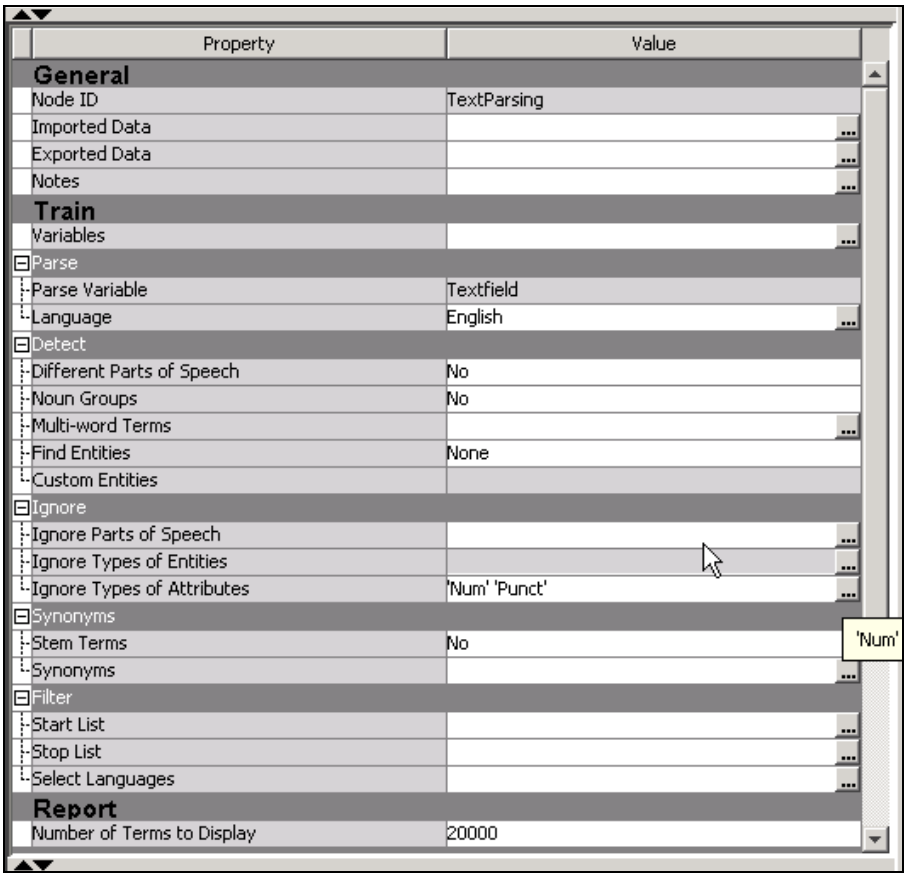
b. Bring in a **File Import** node and import the spreadsheet **Canine_Feline_Optional_Excercise_Chapter3.xls** from inside the D:\workshop\winsas\DMTXT13_1 folder. Run **File Import** and look at the exported data:

EMWS19.FIMPORT_train

Obs #	Document	Textfield
1	1	cat lion tiger cat cat bobcat
2	2	lion leopard jaguar
3	3	dog dog fox wolf jackal jackal dingo
4	4	fox fox coyote dog
5	5	cheetah leopard jaguar cat
6	6	wolf wolf fox fox dog dog coyote
7	7	cat cat cat lion lion lion tiger tiger bobcat bobcat
8	8	dog dog dog dog wolf wolf
9	9	wolf wolf coyote coyote coyote wolf wolf dog
10	10	cheetah cheetah lion lion leopard leopard jaguar jaguar
11	11	cat cat cat cat cat tiger tiger
12	12	dog coyote dog coyote fox fox fox fox
13	13	tiger tiger lion lion lion lion lion
14	14	coyote coyote coyote dog dog dog jackal jackal
15	15	dog dog dingo dingo coyote
16	16	cat lynx panther lynx panther
17	17	lion leopard lynx lynx dog dog dog fox
18	18	cat panther panther tiger tiger tiger fox fox fox coyote

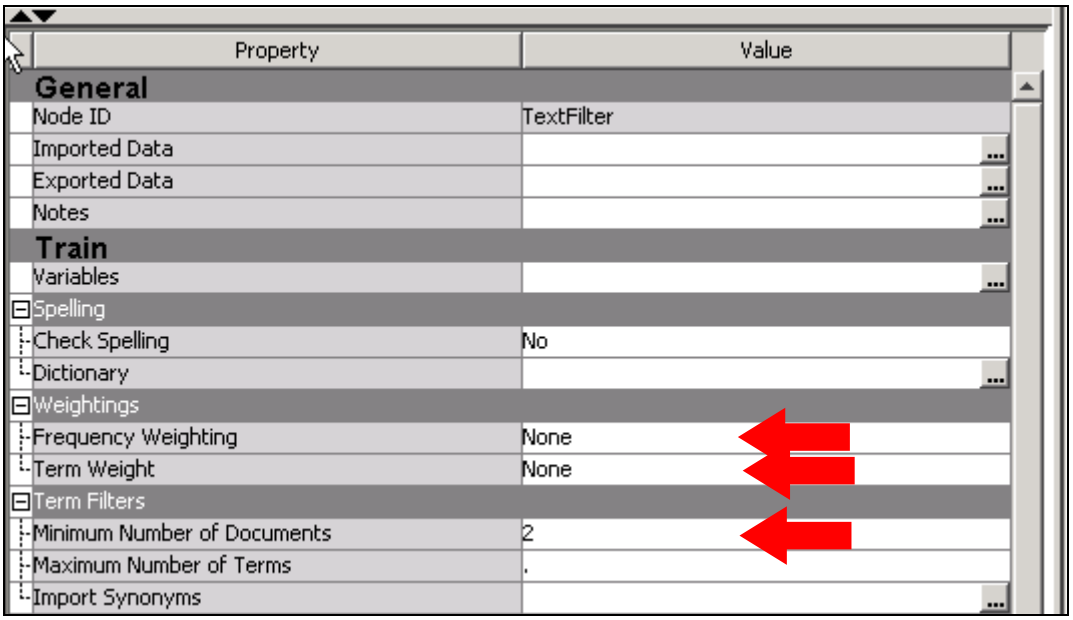
The data consist of 18 records (documents). Each record contains the names of some cat-like (feline) animals and/or some dog-like (canine) animals. If you look carefully, you will see that the first 16 documents contain either pure feline or pure canine animals, but not both. However, in documents 17 and 18, there is a mix of both types of animals.

c. Attach a **Text Parsing** node and change the defaults so that ***all*** the language algorithms are ***turned off***. Your Property Sheet will look like this:



The purpose of turning off all the language algorithms in Text Parsing here (such as Different Parts of Speech, Noun Groups, and no Stop List) is to make this example as simple as possible to follow.

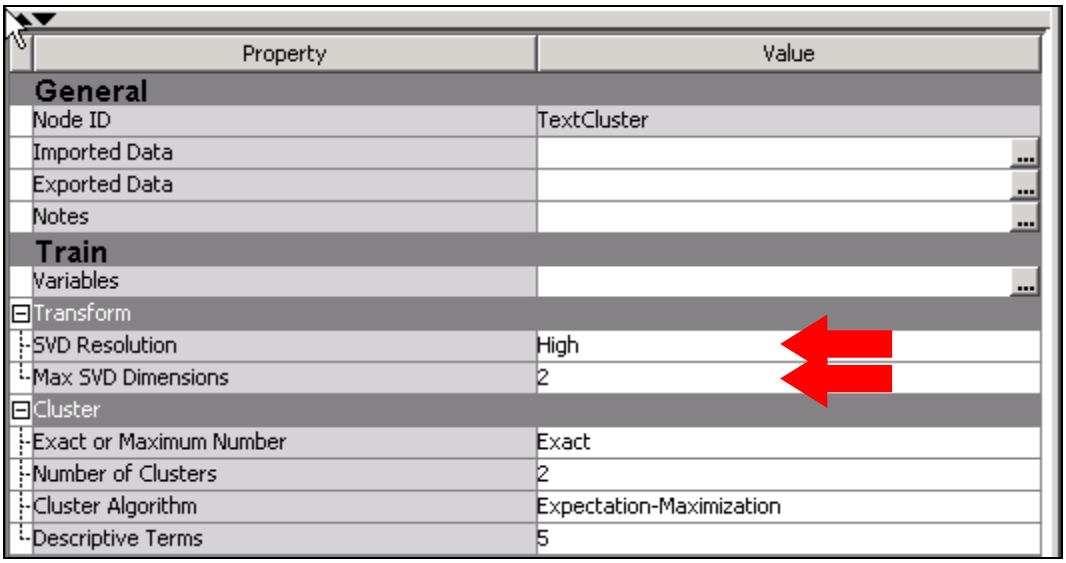
d. Attach a **Text Filter** node to the **Text Parsing** node. Change the defaults on the Property Sheet to those shown below:



Property	Value
General	
Node ID	TextFilter
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Spelling	
Check Spelling	No
Dictionary	
Weightings	
Frequency Weighting	None
Term Weight	None
Term Filters	
Minimum Number of Documents	2
Maximum Number of Terms	.
Import Synonyms	

Note that we are setting the Frequency Weighting and the Term Weight properties to **None**. Again, the reason for this is to construct a simplified example. This will produce a term-document frequency matrix with raw counts rather than weighted counts. Also, remember to change the Maximum Number of Documents property to **2**.

e. Attach a **Text Cluster** node and set up the Property Sheet in this way:



Property	Value
General	
Node ID	TextCluster
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Transform	
SVD Resolution	High
Max SVD Dimensions	2
Cluster	
Exact or Maximum Number	Exact
Number of Clusters	2
Cluster Algorithm	Expectation-Maximization
Descriptive Terms	5

The settings **SVD Resolution=High** and **Max SVD Dimensions=2** will make Text Cluster produce exactly two SVD variables.

f. Run this flow from the Text Cluster node and look at the exported data. You can move the two **TextCluster_SVD** columns by dragging them to the left as shown below.

EMWS19.TextCluster_TRAIN

Obs #	Textfield	TextCluster_SVD1	TextCluster_SVD2
1	cat lion tiger cat cat bobcat	0.33125	-0.94354
2	lion leopard jaguar	0.386678	-0.92221
3	dog dog fox wolf jackal jackal dingo	0.928879	0.370385
4	fox fox coyote dog	0.963237	0.268652
5	cheetah leopard jaguar cat	0.346621	-0.93801
6	wolf wolf fox fox dog dog coyote	0.94154	0.336901
7	cat cat cat lion lion lion tiger tiger bobcat bobcat	0.341098	-0.94003
8	dog dog dog dog wolf wolf	0.910893	0.412644
9	wolf wolf coyote coyote coyote wolf wolf dog	0.913302	0.407282
10	cheetah cheetah lion lion leopard leopard jaguar jaguar ...	0.381565	-0.92434
11	cat cat cat cat cat tiger tiger	0.333646	-0.9427
12	dog coyote dog coyote fox fox fox fox	0.963237	0.268652
13	tiger tiger lion lion lion lion lion	0.365218	-0.93092
14	coyote coyote coyote dog dog dog jackal jackal	0.921711	0.387877
15	dog dog dingo dingo coyote	0.918873	0.394555
16	cat lynx panther lynx panther	0.522318	-0.85275
17	lion leopard lynx lynx dog dog dog fox	0.992874	0.119165
18	cat panther panther tiger tiger tiger fox fox fox coyote	0.823975	-0.56663

Although in most realistic settings, the **TextCluster_SVD** variables are not very interpretable (the document clusters are used for interpretation), in this simple example, you can interpret the results quite easily this way:

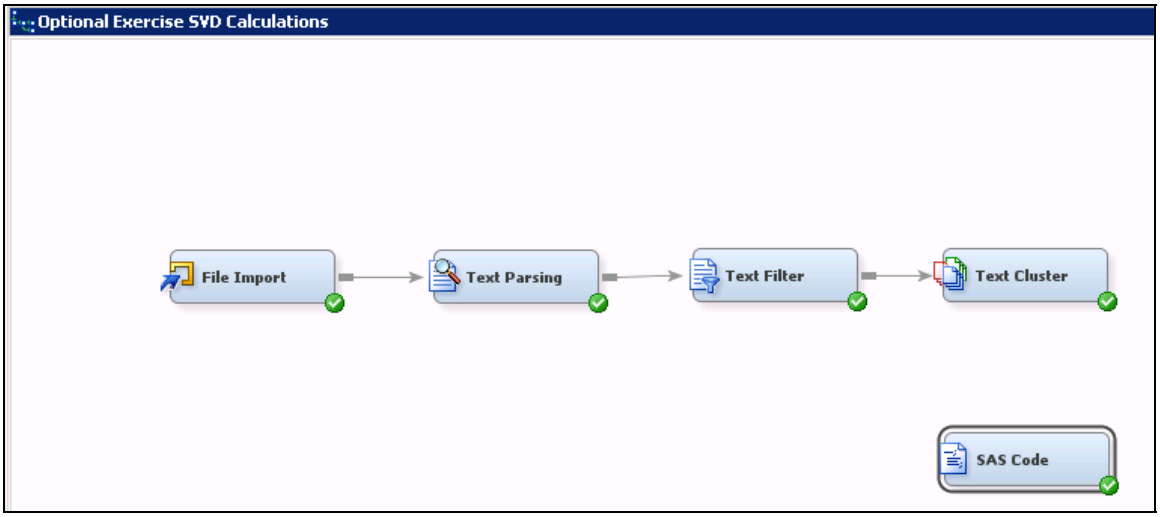
- High **TextCluster_SVD1** values are associated with documents containing the names of canine animals.
- High **TextCluster_SVD2** ***absolute*** values (or very negative values) are associated with documents containing the names of feline animals.

g. Now that you have the **TextCluster_SVD** values generated from the Text Cluster node, you will also obtain them by doing some calculations with PROC IML on the term-document frequency matrix for this example.

With some special SAS data set programming, the 15 (terms) x 18 (documents) term-document frequency matrix for this example could be obtained from the Enterprise Miner project flow. However, in the interests of simplicity, for this exercise you easily obtain the relevant numbers by hand and then verify the entries in the matrix below. These are the raw term-document frequencies (unweighted because the weighting parameters in the Property Sheet of the Text Filter node were turned off). .This is what was called the ***A*** matrix in an earlier slide.

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18
bobcat	1	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
cat	3	0	0	0	1	0	3	0	0	0	5	0	0	0	0	1	0	1
cheetah	0	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0
coyote	0	0	0	1	0	1	0	0	3	0	0	2	0	3	1	0	0	1
dingo	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
dog	0	0	2	1	0	2	0	4	1	0	0	2	0	3	2	0	3	0
fox	0	0	1	2	0	2	0	0	0	0	0	4	0	0	0	0	1	3
jackal	0	0	2	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0
jaguar	0	1	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0
leopard	0	1	0	0	1	0	0	0	0	2	0	0	0	0	0	0	1	0
lion	1	1	0	0	0	0	3	0	0	2	0	0	5	0	0	0	1	0
lynx	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0
panther	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2
tiger	1	0	0	0	0	0	2	0	0	0	2	0	2	0	0	0	0	3
wolf	0	0	1	0	0	2	0	2	4	0	0	0	0	0	0	0	0	0

h. Bring in a **SAS Code** node. There is no need to connect it to any of the other nodes because you will be running only a self-contained PROC IML program. Your diagram should now look like this:



i. Go into the Code Editor on the Property Sheet for the SAS Code node. In the Training Code window. Right-click and select **Open**. Then navigate to the directory **D:\workshop\winsas\DMTXT13_1\sassrc** and then select the program **Proc_IML_Optional_Exercise_Chapter3.sas**. Here is a listing of this program, in two parts to fit across pages):

First part of listing:

```
/***
    PROGRAM="Proc_IML_Optional_Excercise_Chapter3.sas"

    This IML program is self-contained and is given
    to show the matrix calculations used by Text Cluster
    node to produce the 2-dimensional SVD values from the
    "Canine_Feline_Chapter3_Exercise.xls" dataset.

***/
proc iml;
/***
   Matrix A is the unweighted term by document
   frequency matrix for the "Canine_Feline_Chapter3_Exercise.xls"
   dataset.
***/
   A={1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0, /*bobcat*/
      3 0 0 0 1 0 3 0 0 0 5 0 0 0 0 1 0 1, /*cat*/
      0 0 0 0 1 0 0 0 0 2 0 0 0 0 0 0 0 0, /*cheetah*/
      0 0 0 1 0 1 0 0 3 0 0 2 0 3 1 0 0 1, /*coyote*/
      0 0 1 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0, /*dingo*/
      0 0 2 1 0 2 0 4 1 0 0 2 0 3 2 0 3 0, /*dog*/
      0 0 1 2 0 2 0 0 0 0 0 4 0 0 0 0 1 3, /*fox*/
      0 0 2 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0, /*jackal*/
      0 1 0 0 1 0 0 0 0 2 0 0 0 0 0 0 0 0, /*jaguar*/
      0 1 0 0 1 0 0 0 0 2 0 0 0 0 0 0 1 0, /*leopard*/
      1 1 0 0 0 0 3 0 0 2 0 0 5 0 0 0 1 0, /*lion*/
      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0, /*lynx*/
      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2, /*panther*/
      1 0 0 0 0 0 2 0 0 0 2 0 2 0 0 0 0 3, /*tiger*/
      0 0 1 0 0 2 0 2 4 0 0 0 0 0 0 0 0 0}; /*wolf*/

/***
  Call the Singular Value Decomposition routine in order to
  decompose A into its U, Sigma and V matrix factors
***/
  call svd(U,Sigma,V,A);
  print U [format=5.3 label="U Matrix"];
  print Sigma [format=5.3 label="Diagonal of Sigma Matrix"];
  print V [format=5.3 label="V Matrix"];
  print A [format=5.3 label="A Matrix = Term-Doc Frequencies"];
/***
  The next lines keep the first two columns of U matrix
  in order to project documents into a 2-dimensional
  metric space.
***/
  cols_to_keep_U={1 2};
  U_2dim=U[, cols_to_keep_U];
  print U_2dim [format=5.3 label="First 2 Columns of U Matrix"];
```

Second part of listing:

```
/***
  The next lines project the 18 documents into
  the 2-dimensional space.
  The function T(X) used below gives the transpose of the matrix X.
***/
   docs_projected_2dim=T(U_2dim)*A;
/**
  Now transpose the document projections so that we see docs as row vectors.
**/
  docs_projected_2dim_rows=T(docs_projected_2dim);
  print docs_projected_2dim_rows [format = 5.3 label=
  "2 SVD Dimensions for 18 Documents - But Not Yet Normalized"];
/***
   Now normalize document SVD values so that their sums of
   squares = 1 for each document.  This is what Text Cluster does.
***/
  normalized_docs_svd_values=docs_projected_2dim_rows;
  do i=1 to nrow(docs_projected_2dim_rows);
        term1=docs_projected_2dim_rows[i,1]**2;
        term2=docs_projected_2dim_rows[i,2]**2;
        normalize_factor=sqrt(term1+term2);
        normalized_docs_svd_values[i,1]=normalized_docs_svd_values[i,1]/normalize_factor ;
        normalized_docs_svd_values[i,2]=normalized_docs_svd_values[i,2]/normalize_factor;
   end;
   rows={"DOC1"  "DOC2" "DOC3" "DOC4" "DOC5" "DOC6" "DOC7" "DOC8" "DOC9"
         "DOC10" "DOC11" "DOC12" "DOC13" "DOC14" "DOC15" "DOC16" "DOC17" "DOC18"};
   cols={"SVD1" "SVD2"};
   print normalized_docs_svd_values [rowname=rows colname=cols format=8.6 label=
   "Final Normalized Document SVD Values for 2 Dimensions / Compare to Text Cluster Results"];
 quit;
```

j. Run this program in the SAS Code node and look at the results. The final normalized two SVD variables for the 18 documents are at the bottom of the output:

Final Normalized Document SVD Values for 2 Dimensions / Compare to Text Cluster Results

	SVD1	SVD2
DOC1	0.331250	0.943543
DOC2	0.386678	0.922215
DOC3	0.928879	-.370385
DOC4	0.963237	-.268652
DOC5	0.346621	0.938005
DOC6	0.941540	-.336901
DOC7	0.341098	0.940028
DOC8	0.910893	-.412644
DOC9	0.913302	-.407282
DOC10	0.381565	0.924342
DOC11	0.333646	0.942698
DOC12	0.963237	-.268652
DOC13	0.365218	0.930922
DOC14	0.921711	-.387877
DOC15	0.918873	-.394555
DOC16	0.522318	0.852751
DOC17	0.992874	-.119165
DOC18	0.823975	0.566626

These SVD values are the same as the values shown coming out of the Text Cluster node ***except*** for an arbitrary and unimportant sign change for **SVD2**. That is, the **SVD2** values from the Text Cluster node are -1 times the values in this listing.

3.3 Chapter Summary

Text mining consists of the following steps:

1. preparing the data
2. parsing the text
3. converting the text to a numeric representation
4. transforming the numeric representation
5. reducing the dimensionality of the transformed representation
6. analyzing the text through the reduced dimension representation

SAS Text Miner nodes provide numerous strategies for completing the above steps.

The linear algebra approach to text mining, using the singular value decomposition, creates variables (the SVD document vectors) that are used for clustering the documents and for predictive modeling. This approach also reduces the dimensionality of the space of documents. This same approach (modified slightly) generates topics within documents.

For Additional Information

Albright, Russell. 2004. *Taming Text with the SVD*. SAS Institute White Paper.

Albright, R., J.A. Cox, and K. Daly. 2001. "Skinning the Cat: Comparing Alternative Text Mining Algorithms for Categorization." Proceedings of the 2nd Data Mining Conference of DiaMondSUG. Chicago, IL. DM Paper 113.

Cherniak, Eugene. 1993. *Statistical Language Learning.* Cambridge, Massachusetts: The MIT Press.

Evangelopoulos, Nicholas, Xiaoni Zhang, and Victor R. Prybutok. 2010. "Latent Semantic Analysis: Five methodological recommendations." *European Journal of Information Systems*. 1-17.

Jurafsky, Daniel, and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Upper Saddle River, New Jersey: Prentice Hall.

Konchady, Manu. 2006. *Text Mining Application Programming*. Boston: Charles River Media.

Manning, Christopher D., and Hinrich Schutze. 2002. *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts: The MIT Press.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. New York: Cambridge University Press.

Shannon, C.E. 1948. "A Mathematical Theory of Communication." *Bell System Technical Journal*, Vol. 27, pp. 379-423 and 623-656.

Thisted, Ronald A. 1988. *Elements of Statistical Computing*. New York: Chapman and Hall.

Wakefield, Todd. 2004. "A Perfect Storm is Brewing: Better Answers are Possible by Incorporating Unstructured Data Analysis Techniques." *DM Direct*, August 2004.

Wicklin, Rick. 2010. *Statistical Programming with SAS/IML Software.* Cary, NC: SAS Institute Inc.

3.4 Solutions

Solutions to Exercises

1. Comparing the Effect of the Number of SVD Dimensions Using Regression Models

The number of SVD variables actually used by the three regression models was:

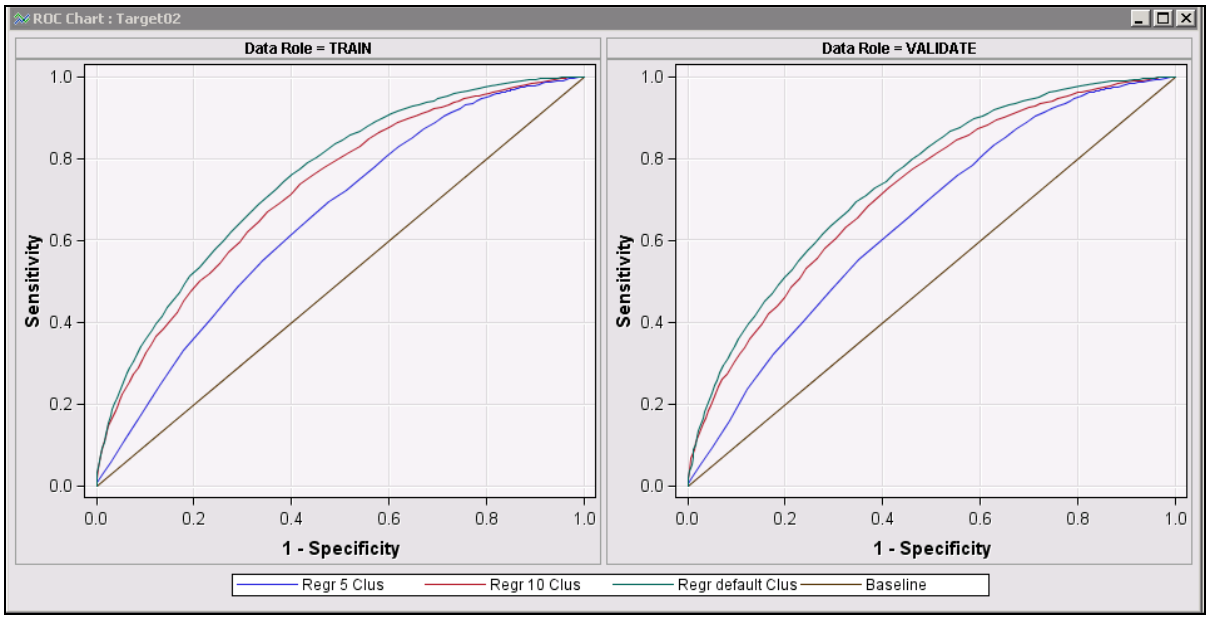
Regr 5 Clus model – 2 SVD variables

Regr 10 Clus model – 8 SVD variables

Regr default Clus model – 26 SVD variables

The easiest way to get these numbers is to go to the Results window for each regression and look at the Effects Plot window. You have to remember to subtract the intercept term to arrive at the values shown above.

Looking at the ROC charts from the Model Comparison tool used to assess the three regression models, it is clear that the three models can easily be rank ordered by their predictive strength.



From the Fit Statistics window, the validation ROC index values are .648, .719, and .744. This confirms that the Regr default Clus model outperformed the other two models and that the default choice of selecting candidate SVD variables led to better results than underspecifying the number of candidate SVD variables as either 5 or 10.

Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Roc Index
Y	Reg	Reg	Regr default Clus	Target02		0.744
	Reg2	Reg2	Regr 10 Clus	Target02		0.719
	Reg4	Reg4	Regr 5 Clus	Target02		0.648

Referring back to the earlier Model Comparison results for the trees and comparing them to their equivalent regression models in terms of their validation ROC index values gives the following:

Model	Validation ROC	Model	Validation ROC
Regr default Clus	.744	DT default Clus	.711
Regr 10 Clus	.719	DT 10 Clus	.706
Regr 5 Clus	.648	DT 5 Clus	.65

The best results are for the **Regr default Clus** model, and this holds up when other fit statistics are looked at, as well.

Solutions to Student Activities (Polls/Quizzes)

3.01 Multiple Choice Poll – Correct Answer

How many documents contain the term *cutting* as indicated in the setup slide?

a. 18
b. 4
c. 14
d. 22

15

3.02 Multiple Choice Poll – Correct Answer

Which term weight is recommended and is the default for documents that are associated with a categorical target variable?

a. IDF
b. mutual information
c. entropy
d. none (G=1)

35

Chapter 4 Additional Ideas and Nodes

4.1 Some Predictive Modeling Details

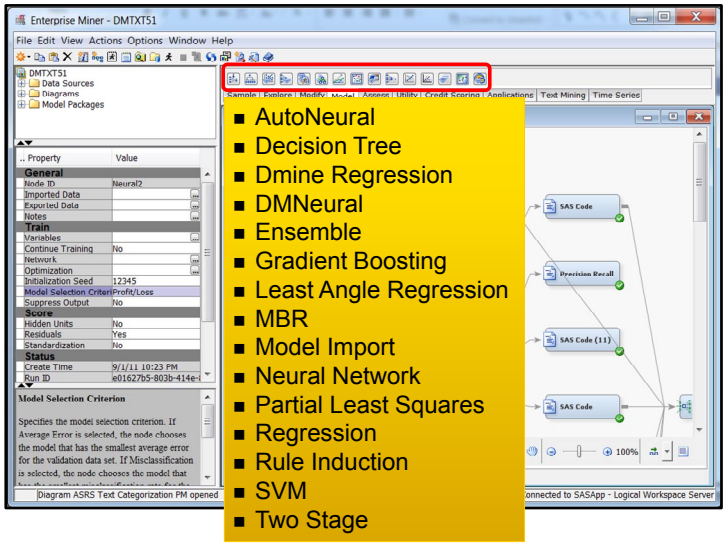
Objectives

- Describe predictive modeling data sets.
- Explain predictive modeling projects and features of SAS Enterprise Miner related to predictive modeling.
- Explain the trade-off between predictive power and interpretability.
- Discuss how the Text Cluster and Text Topic nodes can be set up to affect this trade-off.
- Emphasize the need for experimenting with different predictive modeling and text miner settings—that is, the “workbench” idea for Enterprise Miner.

3

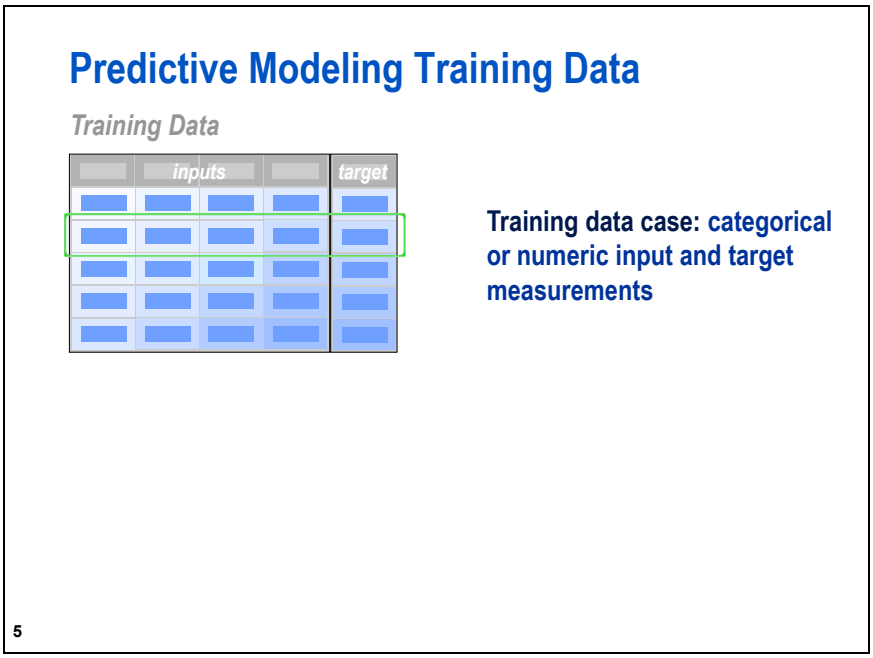
SAS Enterprise Miner has many predictive model nodes. Some nodes are general purpose, such as the Decision Tree, Neural Network, and Regression nodes. Some nodes are specialized, such as the MBR, Rule Induction, and Partial Least Squares nodes. This course has illustrated the use of the Decision Tree and Regression nodes. The availability of many different modeling techniques makes it easy to try out different approaches to find the best results for your data. Think of the Enterprise Miner and its many different nodes as a ***workbench for analytic experimentation***.

SAS Enterprise Miner Model Tab

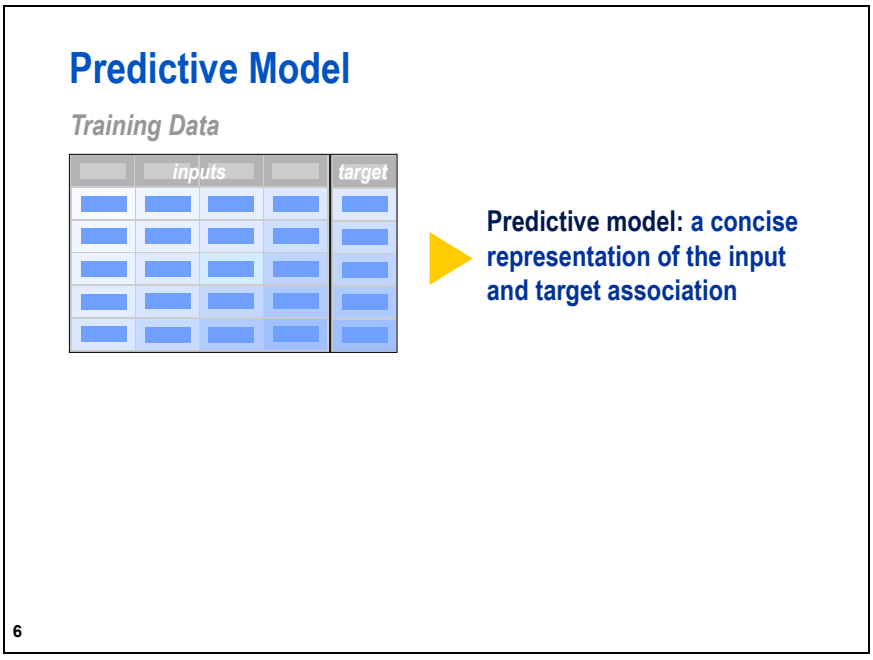


- AutoNeural
- Decision Tree
- Dmine Regression
- DMNeural
- Ensemble
- Gradient Boosting
- Least Angle Regression
- MBR
- Model Import
- Neural Network
- Partial Least Squares
- Regression
- Rule Induction
- SVM
- Two Stage

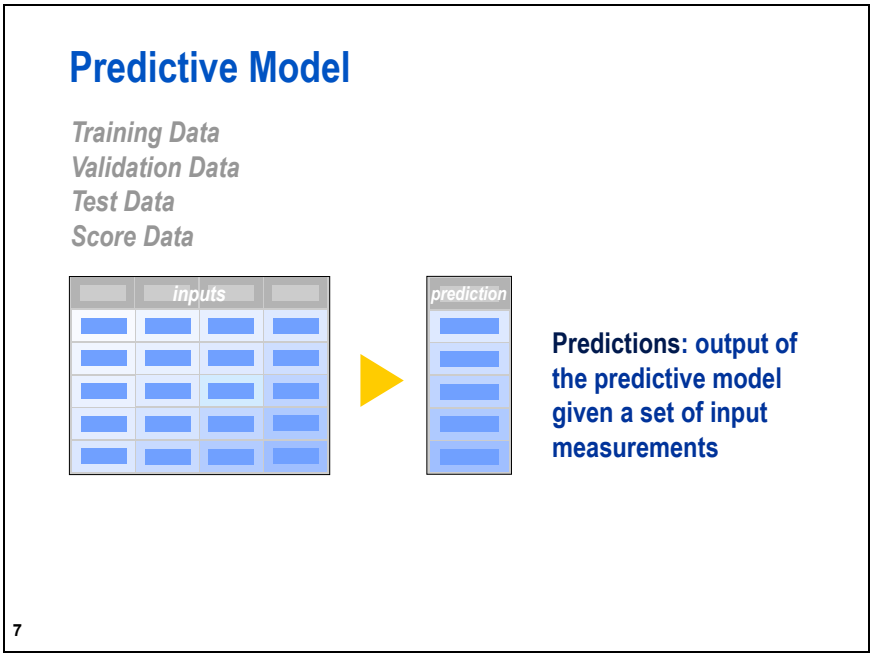
4



The minimum requirement for data mining predictive modeling is at least one target variable and at least one input variable. A predictive model is constructed using a training data set. The model attempts to predict the value of the target variable using only the values of a set of input variables. For example, input variables can measure customer attributes such as gender, age, income, location of primary residence, and average purchases to try to estimate the probability that a customer will respond to a particular promotion, such as a 20% off discount on purchases of $100 or more.

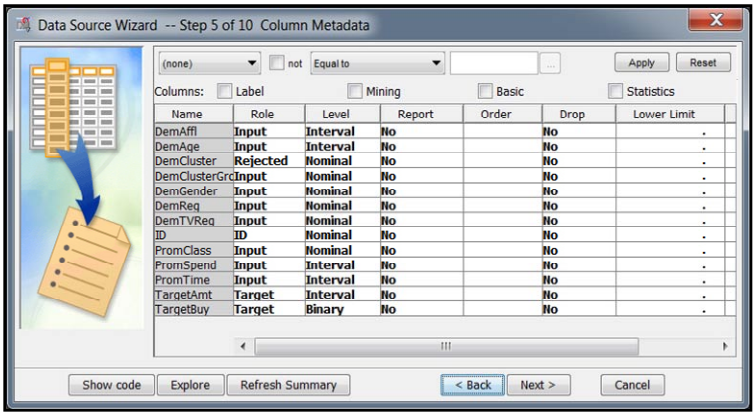


After a model is constructed using training data, the performance of the model can be assessed using a holdout data set. When a final model is selected, it can be used to score new data to determine, for example, which customers should be selected for a promotional offer. The term *score* is synonymous with *predict*.



To choose from a variety of models, a holdout data set called a *validation* data set is used to determine how well models will extrapolate to new data. This helps overcome the problem of *overfitting*, which occurs when a model is constructed to fit the training data set so well that it does not fit any other data well. For a model that has been selected for deployment, a second holdout data set, called a *test* data set, is used to get an unbiased estimate of the accuracy of the model in the live environment. A predictive model can score any data set that has the inputs used by the model. It is important to ensure that models are applied to data commensurate with how a model was constructed. For example, a model constructed using only customers who reside in California might not be appropriate for scoring customers in Florida.

SAS Enterprise Miner Source Data



For predictive modeling:

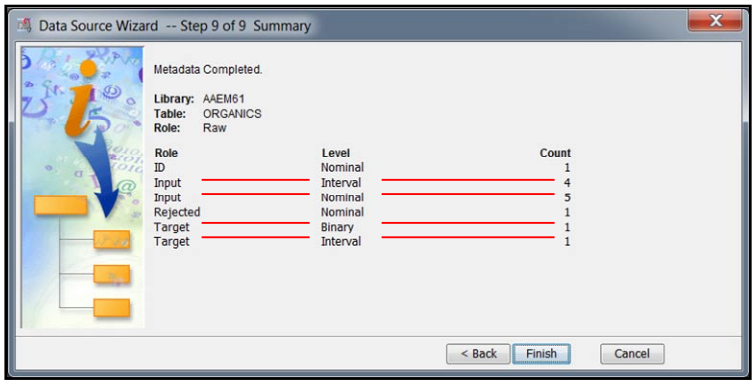
- at least one variable with the role ***Target***
- at least one variable with the role ***Input***

8

On the slide above, the table has nine input variables that will be used as candidate predictor variables and two target variables. The two target variables were automatically recognized because they contain the prefix *Target*. One of the target variables should be selected for modeling and the other one rejected.

When you create a data source in SAS Enterprise Miner, the Data Source Wizard displays all the variable roles so that you can check that the target and input variables have been specified in how you need to build a predictive model.

SAS Enterprise Miner Source Data

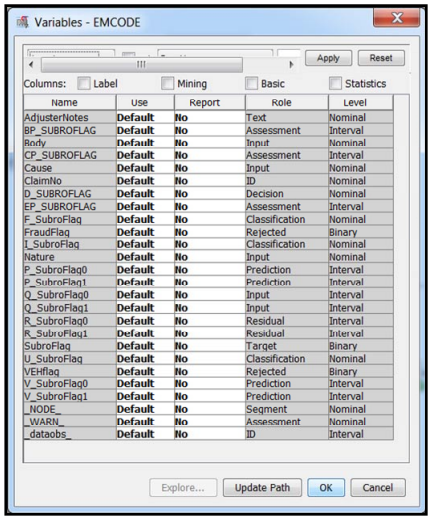


The Data Source Wizard summarizes the metadata.

There are nine inputs and two targets.

9

SAS Enterprise Miner Predictions Data

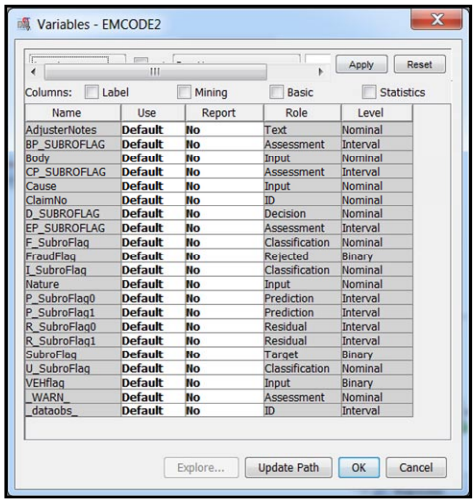


Decision Tree

10

A predictive model will provide a predicted value. In the case of a binary target variable, the predicted value is the posterior probability of the primary event given the inputs. You can use this probability to derive a decision rule. (For example, if the probability exceeds 0.37, send the promotion to the customer. Otherwise, do nothing.) SAS Enterprise Miner model nodes will add a variety of columns to the imported data when creating the exported data. The nature of the added columns depends on the model used. The above table is displayed by selecting the **Variables** property in a SAS Code node attached to a Decision Tree node. The predicted value in this case is called **P_SubroFlag1**. This is the probability that the target variable (**SubroFlag**) has the value 1. There is also a variable named **P_SubroFlag0**, which is the complementary value, 1-**P_SubroFlag1**.

SAS Enterprise Miner Predictions Data



Regression

11

Prediction Types for Binary Response Models

Decisions:

- I_Target is 1 if P_Target1>P_Target0. It is 0 (zero) otherwise. Thus, I_Target decisions are equivalent to using a posterior probability cutoff of 50%.
- D_Target is 1 if Profit(Target=1)>Profit(Target=0). It is 0 (zero) otherwise. If no profit information is provided, then D_Target is equivalent to I_Target.

Estimates:

- P_Target1 is the estimated posterior probability that Target=1.
- P_Target0 is the estimated posterior probability that Target=0.

12

The above slide summarizes the prediction variables that are usually of interest. A Decision Tree node would produce, for example, **P_Target1**, which would be the posterior probability derived from the tree after correcting for oversampling. Note that if the binary target variable has the name **FLOYD**, then the posterior probability that **FLOYD**=1 is given by the variable **P_FLOYD1**.

SAS Enterprise Miner Input Selection

- Explore Tab
  - Variable Clustering node
  - Variable Selection node
- Model Tab
  - Decision Tree node
  - Regression node

These can be used with all the variables created by the Text Cluster and Text Topic nodes.

13

Although lack of input variables can be a problem, frequently your problem is that you have a very large number of candidate input variables to select from. Input selection, or variable selection, is an important topic in predictive modeling. SAS Enterprise Miner facilitates input selection in a number of different ways.

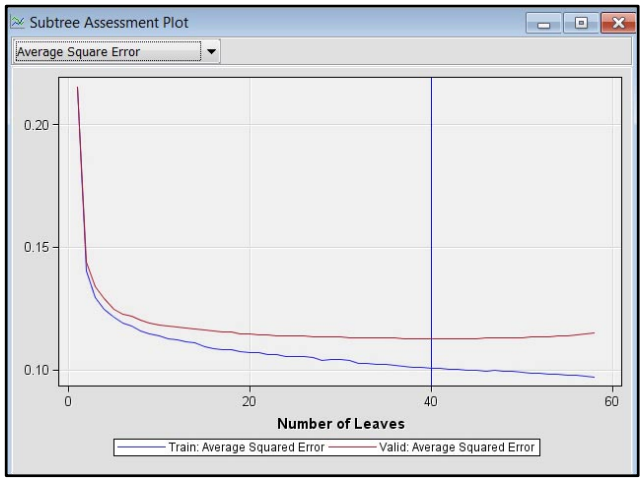
SAS Enterprise Miner Dimensionality Reduction

- Explore Tab
  - Variable Clustering node
- Modify Tab
  - Principal Components
- Model Tab
  - Partial Least Squares node

14

A Decision Tree node performs input selection by deciding which subset of input variables will be used to partition the data into separate leaves. If a variable is not useful in separating the data into more pure leaf nodes, then the variable is discarded. In the above plot, a tree with 40 leaves is derived. A 59-leaf tree was pruned to remove leaves that did not improve overall model accuracy. The pruned subtree is used to choose the input variables that are useful for prediction. These variables will be passed to successor nodes, whereas the other input variables will have their roles changed to **Rejected**.

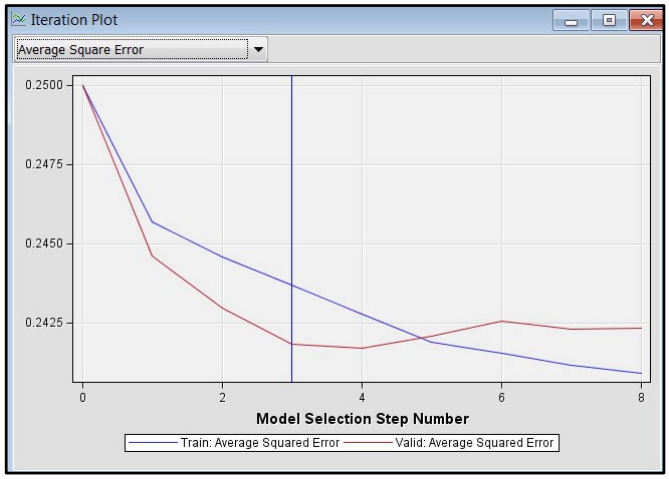
Model Selection=Input Selection



Decision Tree Subtree Assessment Plot

15

Model Selection=Input Selection



Regression Iteration Plot

16

The Regression node also has options for performing variable selection. The above iteration plot reveals that the Regression node tried a series of models, culminating in an eight-variable model. But for a final model, it chose one that has only three variables.

Model Assessment

- Fit Statistics
  - Average square error
  - Misclassification rate
  - Information criteria
  - Others
- Charts and Plots
  - ROC chart
  - Gains chart
  - Lift chart
  - Others

17

Model assessment is performed using results from the model nodes and the Model Comparison node.

Model Assessment: Fit Statistics

Fit Statistics

Target	Fit Statistics	Statistics Label	Train	Validation	Test
Target19	_NOBS_	Sum of Frequencies	16139	5380	.
Target19	_MISC_	Misclassification Rate	0.13204	0.150743	.
Target19	_MAX_	Maximum Absolute Error	0.975649	0.975649	.
Target19	_SSE_	Sum of Squared Errors	3250.704	1213.841	.
Target19	_ASE_	Average Squared Error	0.10071	0.112811	.
Target19	_RASE_	Root Average Squared Error	0.317348	0.335873	.
Target19	_DIV_	Divisor for ASE	32278	10760	.
Target19	_DFT_	Total Degrees of Freedom	16139	.	.

Decision Tree Node Fit Statistics Table

18

Fit statistics, such as average squared error (ASE), are defined in the SAS Enterprise Miner documentation, which you can also access by selecting **Help ⇨ Contents.**

Model Assessment: Fit Statistics

Fit Statistics

Target	Fit Statistics	Statistics Label	Train	Validation	Test
Target19	_DFT_	Total Degrees of Freedom	16139	.	.
Target19	_DFE_	Degrees of Freedom for Error	15985	.	.
Target19	_DFM_	Model Degrees of Freedom	154	.	.
Target19	_NW_	Number of Estimated Weights	154	.	.
Target19	_AIC_	Akaike's Information Criterion	9423.025	.	.
Target19	_SBC_	Schwarz's Bayesian Criterion	10607.13	.	.
Target19	_ASE_	Average Squared Error	0.08377	0.096703	.
Target19	_MAX_	Maximum Absolute Error	0.994556	0.993259	.
Target19	_DIV_	Divisor for ASE	32278	10760	.
Target19	_NOBS_	Sum of Frequencies	16139	5380	.
Target19	_RASE_	Root Average Squared Error	0.28943	0.310971	.
Target19	_SSE_	Sum of Squared Errors	2703.924	1040.523	.
Target19	_SUMW_	Sum of Case Weights Times Freq	32278	10760	.
Target19	_FPE_	Final Prediction Error	0.085384	.	.
Target19	_MSE_	Mean Squared Error	0.084577	0.096703	.
Target19	_RFPE_	Root Final Prediction Error	0.292205	.	.
Target19	_RMSE_	Root Mean Squared Error	0.290821	0.310971	.
Target19	_AVERR_	Average Error Function	0.282391	0.320264	.
Target19	_ERR_	Error Function	9115.025	3446.043	.
Target19	_MISC_	Misclassification Rate	0.111593	0.129554	.
Target19	_WRONG_	Number of Wrong Classifications	1801	697	.

Neural Network Node Fit Statistics Table

19

Different models produce different fit statistics.

Model Assessment: Charts and Plots

Model Comparison Node ROC Chart

20

The Model Comparison node produces additional fit statistics and charts and plots that allow the direct comparison of the performance of different models. Some plots are not available directly through the GUI, but SAS Enterprise Miner provides extensive functionality through the SAS Code node.

In this class, we have focused on using text mining tools to derive new input variables from free-form text data. Therefore, it is valuable to make users of these tools aware of some of the strategies and "tricks" that are used that can enhance your predictive models.

An important point to understand is that there is usually a trade-off between predictive power and the interpretability of a model. That is, if you try to obtain the most powerful predictive model that you can, you often end up with a more complicated and less interpretable model. On the other hand, for many purposes, it is often important to obtain models that can be explained and understood by others, including senior management and clients who will be using the model.

A practical strategy for handling this trade-off is to create ***both*** types of models: one that has the strongest predictive power that you can obtain and another one that is more explainable. Then you can present both of these models to your audience, and they can participate in the decision of which one to use. This requires a fair amount of experimenting with settings and also understanding what choices you have when you use the Text Cluster and Text Topics nodes. These choices can directly make your model more likely to have greater predictive power ***or*** more easily interpretable.

A Trade-Off: Models with Greater Predictive Power versus Greater Interpretability

- The many tools available in the Enterprise Miner, including the Text Mining nodes, provide analysts with lots of ways to explore this trade-off with their models.
- A very reasonable approach to handle the trade-off is to
  - obtain the strongest model that you can regardless of its interpretability
  - obtain a more interpretable model at (possibly) the expense of predictive power
  - present both models to your audience (senior management, clients) and let them choose between the two.

21

As we saw earlier in Chapter 1, there are several variables that are produced by the Text Cluster and Text Topic nodes. Text Cluster generates the document SVD vectors, which are then used to cluster the documents. The SVD variables are automatically set to the role of **Input**, so they are ready to be used for a prediction model. However, the actual clusters that are produced by these SVD variables are set to the role of **Segment** and therefore are not immediately available for prediction purposes. Nevertheless, all that is required to make this change is to convert the **TextCluster_cluster_** variable from the role of **Segment** to the role of **Input**. Because the **TextCluster_cluster** variable is intended to help the analyst in interpreting results, many analysts will consider using the **TextCluster_cluster_** variable for modeling purposes instead of the SVD variables. ***This is a deliberate trade-off that might mean sacrificing some predictive power for greater interpretability.***

The Trade-Off with Text Cluster Variables

- You should be familiar with one strategy to address this trade-off when using the Text Cluster node.
- The Text Cluster node generates up to three sets of variables:
  - the document raw SVD variables (**TextCluster_SVD** is always generated.)
  - the document cluster variable (**TextCluster_cluster_** is always generated.)
  - the probabilities of a document belonging to each cluster (**TextCluster_prob** is generated only when the E-M clustering algorithm is used.)
- By default, only the raw SVD variables will be used for modeling, but these, in actual practice, are not likely to be interpretable.

22

Another “trick” that analysts sometimes use in prediction modeling is to use the Text Cluster node segment probabilities as input variables. These are the **TextCluster_prob*N*** variables that are created when the Expectation-Maximization clustering algorithm is used. Using this algorithm, if there are three clusters created (so **TextCluster_cluster**=1, 2, or 3), then a document will have a probability **TextCluster_prob1**, **TextCluster_prob2**, and **TextCluster_prob3** associated with each of the clusters. The document will be assigned to the cluster for which it has the highest probability, but these probabilities can be directly used as input variables if you want.

In either case, if you want to swap in the **TextCluster_cluster_** or **TextCluster_prob** variables, you will have to use a Metadata node to do this. You would then reject the **TextCluster_SVD** variables so that they are not used in the analysis.

The Trade-Off with Text Cluster Variables

- On the other hand, the **TextCluster_cluster_** variable is designed to be interpretable using the descriptive terms associated with each cluster.
- Similarly, the **TextCluster_prob** variables can be very helpful for understanding results (when the E-M algorithm is used). For example, suppose there are three clusters with these descriptive terms:
  - *zoo*, *lion*, … (cluster 1 – animal related)
  - *baseball*, *soccer*, … (cluster 2 – sports related)
  - *weather*, *hot*, … (cluster 3 – weather related)
- If a document has probabilities of belonging to these clusters (using the E-M algorithm), respectively, of .50, .48, .02, it will be assigned to cluster 1, but it is likely to be sports related, as well. (cluster 2)

23

The Trade-Off with Text Cluster Variables

- An analyst can build more interpretable predictive models (perhaps sacrificing some predictive power) by excluding the raw SVD and using either the **TextCluster_cluster_** variable or the **TextCluster_prob** variables as inputs to a model.
- This would be done through the Metadata node.

24

An analyst can also manage the Text Topic node in a way that can trade off between better predictive power or clearer interpretability.

The Trade-Off with Text Topic Variables

- There is a somewhat similar strategy available for the predictive power versus interpretability trade-off when using the Text Topic node.
- As we have discussed, the Text Topic nodes generate two types of variables:
  - The **TextTopic_raw** variables, which are continuous measures that indicate the strength of a topic present in a document.
  - The **TextTopic** variables, which are binary measures that indicate whether a document has a topic or not, based on using a cut-off value for the **TextTopic_raw** variable.

25

The Trade-Off with Text Topic Variables

- Both types of variables are always generated, but by default, the **TextTopic_raw** variables are set to **Input** for predictive modeling whereas the **TextTopic** binary variables are not.
- Both sets of variables can be interpreted with the same key terms that describe the topic.
- However, some analysts find the binary **TextTopic** variables easier to understand because a document is classified as either having a topic or not.
- Evidently, you sacrifice some information when using the **TextTopic** variables instead of the **TextTopic_raw** variables, but this can lead to a more interpretable model.

26

Experiment!

- Regardless of whether you are aiming for better predictive power or greater interpretability, you should adopt an experimental approach to modeling.
- Enterprise Miner is like a workbench where you can easily try out different approaches and compare their results.
- The default settings are meant to work well across a wide variety of situations, but ***your*** particular analytic problem can very often be improved by testing out different parameter settings.
- ***Do not*** take the defaults for granted as always producing the best results.

27

Experimenting with the Effects of Global Weights on Predictive Power

This demonstration further shows how to approach text mining analytics and predictive modeling experimentally by trying out different parameter settings or different modeling techniques. In this case, we experiment with different term weight (global weight) settings and use the Model Comparison node to compare the models. These results produce somewhat surprising results regarding the weights and provide motivation for trying out different approaches.

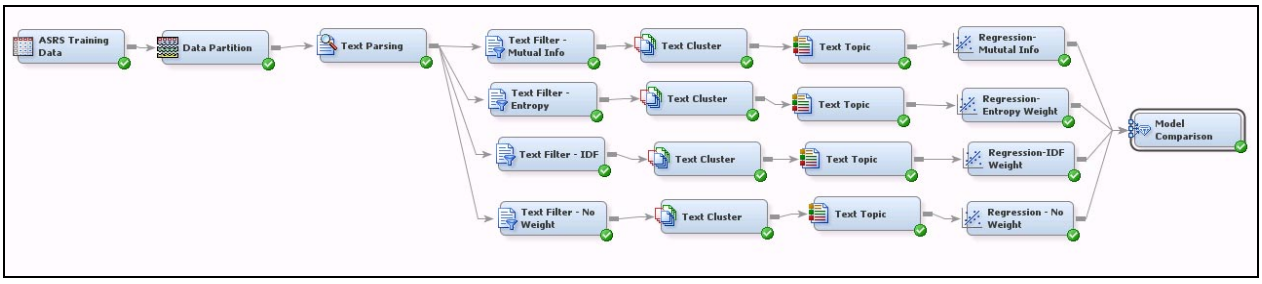
We will again use the **ASRS** data set. This time, though, change the target from **Target02** (which has to do with noncompliance events and was used in the earlier demonstration of Chapter 3) to **Target05** (which has to do with the occurrence of a collision hazard event).

The 22 target events vary considerably with respect to the difficulty of modeling them. Descriptions of several of these target events (as given in E.G. Allan et al 2008), along with published ROC index values for models that Allan et al obtained using an analytic method known as Nonnegative Matrix Factorization, are shown in the table below. Note that we were able to improve on their ROC index for **Target02** in the last chapter, where we obtained .711 with a decision tree using the default SVD variables generated from the Text Cluster node. In this demonstration, using **Target05**, we are also able to improve on the reported Allan et al results.

Some of the 22 ASRS target events with their ROC index values from published model results:

Event Label in Course Data	Description of Event	Reported ROC Index From Allan et al Model Results
Target02	Operation Noncompliance	.6009
Target05	Incursion (collision hazard)	.8977
Targer13	Weather Issue	.6287
Targer21	Illness or Injury event	.8201
Target22	Security concern / threat	.9040

We will create a diagram flow that tests out the effect of trying the four different global weight (term weight) options available in the Text Filter node. The flow will look like this when finished:



1. Create a diagram named **Global Weight Experiment**. Drag in the **ASRS** training data. This time make **Target05** the target variable, with all the other targets rejected:

Name	Role	Level
ID	**ID**	**Nominal**
Size	**Rejected**	**Interval**
Target01	**Rejected**	**Binary**
Target02	**Rejected**	**Binary**
Target03	**Rejected**	**Binary**
Target04	**Rejected**	**Binary**
Target05	**Target**	**Binary**
Target06	**Rejected**	**Binary**
Target07	**Rejected**	**Binary**
Target08	**Rejected**	**Binary**
Target09	**Rejected**	**Binary**
Target10	**Rejected**	**Binary**
Target11	**Rejected**	**Binary**
Target12	**Rejected**	**Binary**
Target13	**Rejected**	**Binary**
Target14	**Rejected**	**Binary**
Target15	**Rejected**	**Binary**
Target16	**Rejected**	**Binary**
Target17	**Rejected**	**Binary**
Target18	**Rejected**	**Binary**
Target19	**Rejected**	**Binary**
Target20	**Rejected**	**Binary**
Target21	**Rejected**	**Binary**
Target22	**Rejected**	**Binary**
Text	**Text**	**Nominal**

2. Bring in the **Data Partition** node and leave it at the default settings of Training/Validation/Test=40/30/30.

3. Connect the **Data Partition** node to a default **Text Parsing** node.

4. The **Text Parsing** node is then connected to each of four individual **Text Filter** nodes. Each of these Text Filter nodes will be set up using a different term weight (global weight):

   a. The first Text Filter is set to **Mutual Information**.

   b. The second Text Filter is set to **Entropy**.

   c. The third Text Filter is set to **IDF**.

   d. The fourth Text Filter is set to **None**.

   Remember that **Mutual Information** would be the default here because a target variable (**Target05**) has been defined.

   Rename each of the four Text Filter nodes to identify which global weight has been used, such as **Text Filter - Mutual Info** and **Text Filter - IDF**.

5. Connect each **Text Filter** node in the series to its own default **Text Cluster** node and then its own **Text Topic** node.

6. The output of each **Text Topic** node is then connected to a **Regression** node and the following settings are selected:

Model Selection	
Selection Model	Forward
Selection Criterion	Validation Error
Use Selection Defaults	Yes

7. Rename each of the Regression nodes to indicate the term weight that was used earlier by its Text Filter node—that is, **Regression-Mutual Info**, **Regression-Entropy**, and so on. (The reason for renaming each of these Regression nodes is that it will make comparing results easier when we look at the Model Comparison node.)

8. Connect each individual **Regression** node to a single **Model Comparison** node. Change the Model Comparison node to make the ROC index on the test data the selection statistic:

Model Selection	
Selection Data	Default
Selection Statistic	ROC
Grid Selection Statistic	Default
Selection Table	Test
Selection Depth	10

9. Check to see that you have set things up as shown in the display capture at the beginning of this demonstration and then run the entire flow from the Model Comparison node.

   If you are running this flow from scratch, it will take about 12 or 13 minutes using the Virtual Machine image provided for this class.

10. Open the Model Comparison results and look at the Fit Statistics window to compare the performance of the four models using different term weights:

Model Description	Target Variable	Target Label	Selection Criterion: Test: Roc Index
Regression-IDF Weight	Target05		0.963
Regression-Entropy Weight	Target05		0.962
Regression - No Weight	Target05		0.952
Regression-Mututal Info	Target05		0.929

First, note that all the models are producing very high ROC index values on the test data, as well as on the training and validation data sets. This is also obvious from the dramatically high curves in the ROC Chart for **Target05**. Other statistics tell the same story. So in general, we have been quite successful at classifying ASRS reports as either indicating a collision hazard event or not.

What is very surprising, however, is that the **Mutual Information** term weight in this case produced the ***worst*** results (Index=.929), even worse than using no weight (Index=.952).

The lesson here is to be experimental and try out different parameter settings! The default settings generally work well, but the analysis of each data set should be explored in a number of ways for best results.

4.01 Short Answer Poll

What are the Test Average Squared Error values for the four models generated with different term weights in the demonstration?

29

Exercises

1. **Adding a Decision Tree Node to the Flow to Compare to the Previous Four Regression Models**

   a. Add a **Decision Tree** node to the flow used in the demonstration. Connect this to the **Text Topic** node that is part of the flow using the **IDF** term weight (because we previously found this term weight to produce the best results).

   b. Rename the tree **to Decision Tree - IDF**.

   c. Make two changes in the tree default settings. First, set **Leaf Size** to **25** and then set **Assessment Measure** to **Average Square Error**.

   d. Connect the tree to the **Model Comparison** node and run it from there.

   e. How does the tree compare to the other four models in terms of the ROC index on the test data?

4.2 Text Rule Builder Node

Objectives

- Provide introductory details about the Text Rule Builder node.
- Identify the property settings of Text Rule Builder node.
- Perform predictive modeling using the Text Rule Builder node.

33

Text Rule Builder Node: Introduction

- The Text Rule Builder node provides a stand-alone predictive modeling solution for data having a text variable and a categorical target variable.
- This node generates an ordered set of rules that together are useful in describing and predicting a target variable.
- This node facilitates “active learning” in which a user can dynamically interact with an algorithm to iteratively build a predictive model.
- This node creates Boolean rules from small subsets of terms to predict a categorical target variable.
- The node must be preceded by Text Parsing and Text Filter nodes.

34

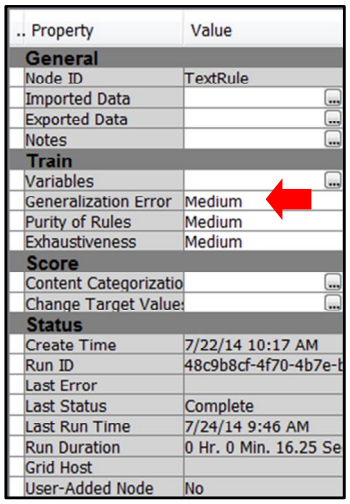
Text Rule Builder Node: Introduction

- The node must have a target variable with a measurement level of binary, ordinal, or nominal.
- The Partitioning node should appear before the Text Parsing node, and validation or test data sets must contain the same target variable identified in the training data.

35

Text Rule Builder Node Train Properties

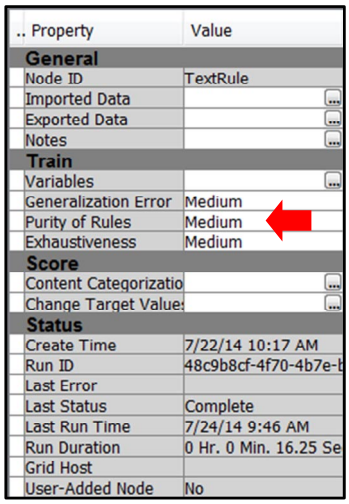
- The Generalization Error property determines the predicted probability for rules that use an untrained data set (validation).
- This is to prevent overtraining. Higher values do a better job of preventing overtraining at a cost of not finding potentially useful rules.



36

Text Rule Builder Node Train Properties

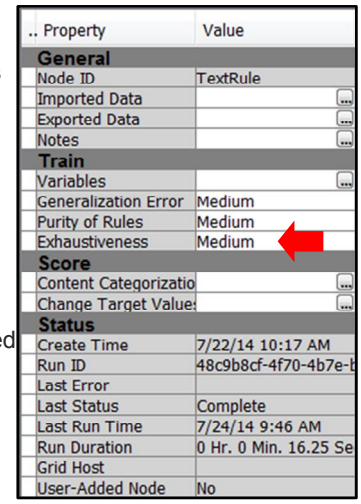
- The Purity of Rules property controls the complexity of rules to consider.
- Purity of Rules determines how selective each rule is by controlling the maximum *p*-value necessary to add a term to a rule.
- Selecting **Very High** results in the fewest, purest rules. A rule with only a few words is considered to be purer than a rule with many words.
- Selecting **Very Low** results in the most rules that handle the most terms.
- Valid values are Very Low (p<.17), Low (p<.05), Medium (default, p<.005), High (p<.0005), and Very High (p<.00005).



37

Text Rule Builder Node Train Properties

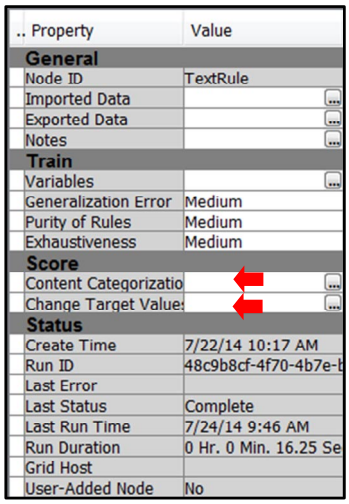
- Exhaustiveness determines the exhaustiveness of the rule search process, or how many potential rules are considered at each step.
- As you increase the exhaustiveness, you increase the amount of time that the Text Rule Builder node requires and increase the probability of overtraining the model.
- A high value permits many rules. Unlike the other two properties, a high value increases the time required to find useful rules. For the other two properties, high values decrease the time required to find useful rules.
- Valid values are Very Low, Low, Medium (default), High, and Very High.



38

Text Rule Builder Node Score Properties

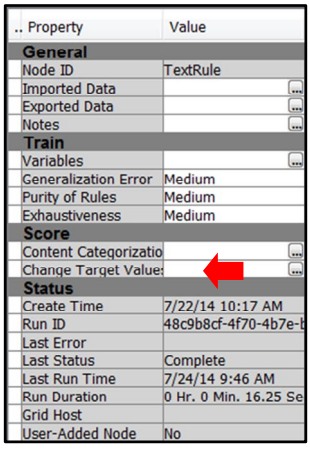
- **Content Categorization Code –** Click the ellipsis button to the right of the Content Categorization Code property to view the Content Categorization Code window. The code that is provided in this window can be copied and pasted into SAS Content Categorization Studio. This node must be run before you can access the Content Categorization Code window.
- **Change Target Values** – Click the ellipsis button to the right of the Change Target Values property to view the Change Target Values window. The window enables you to view and reassign target values. As a result, you can rerun the Text Rule Builder node and iteratively refine your model.



39

Text Rule Builder Node Score Properties

- **Change Target Values**: This window enables you to view and reassign target values. Thus, you can rerun the Text Rule Builder node and iteratively refine your model prediction.
- This property setting facilitates *active learning*, in which a user can dynamically interact with an algorithm to iteratively build a predictive model.
- The observations in this window contain all observations in the training, validation, or test data set that meet any of the following conditions:
  - all misclassified observations
  - includes observations for which the target contains a missing value
  - an observation for which you have previously changed the imported target value to a different target value
- The observations in this window are ordered by the model's *posterior probability* in descending order from 1 to 0.



40

Predictive Modeling Using the Text Rule Builder Node

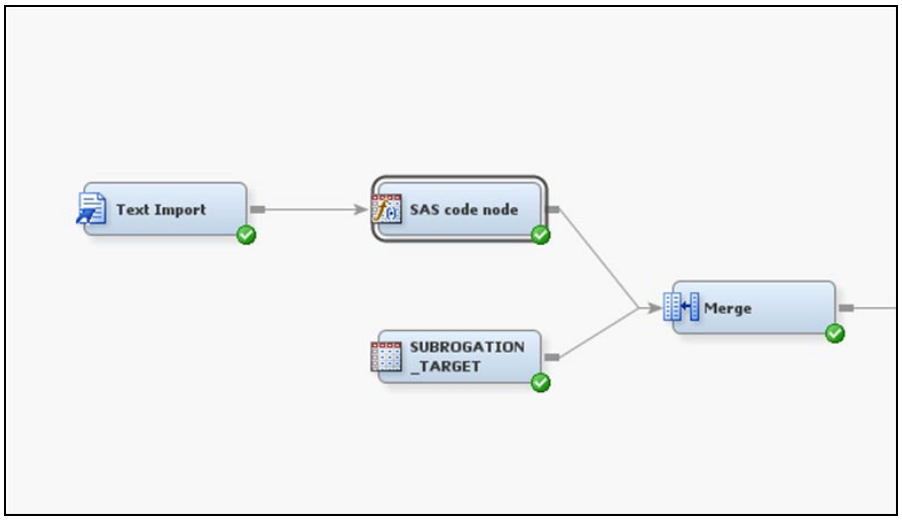
This demonstration illustrates the capabilities and applications of the Text Rule Builder node in creating Boolean rules from small subsets of terms to predict Workers' Compensation insurance recoveries.

The demonstration uses the Predicting Workers' Compensation Recovery Potential data source generated in Chapter 2.

Follow the following steps to create a new predictive modeling based on Rule Builder:

1. Open the **Subrogation Model** diagram and select the first four nodes, including the **Merge** node, and copy it to the clipboard.

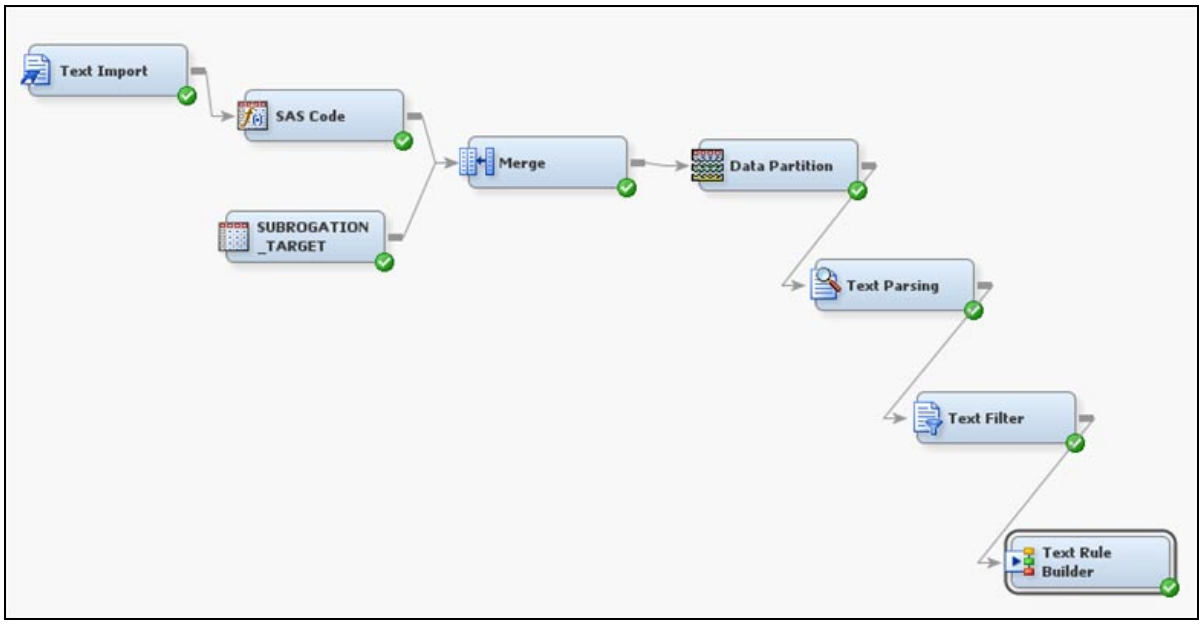
2. Create a new diagram named **Subrogation Rule Builder** and paste the four copied nodes into the blank workspace.



3. Connect a **Data Partition** node (from the Sample tab) to the **Merge** node and change the Data Set Allocation settings to Training=:60, Validation=:40, and Testing:=0. This splits the data set into 60% training and 40% validation.

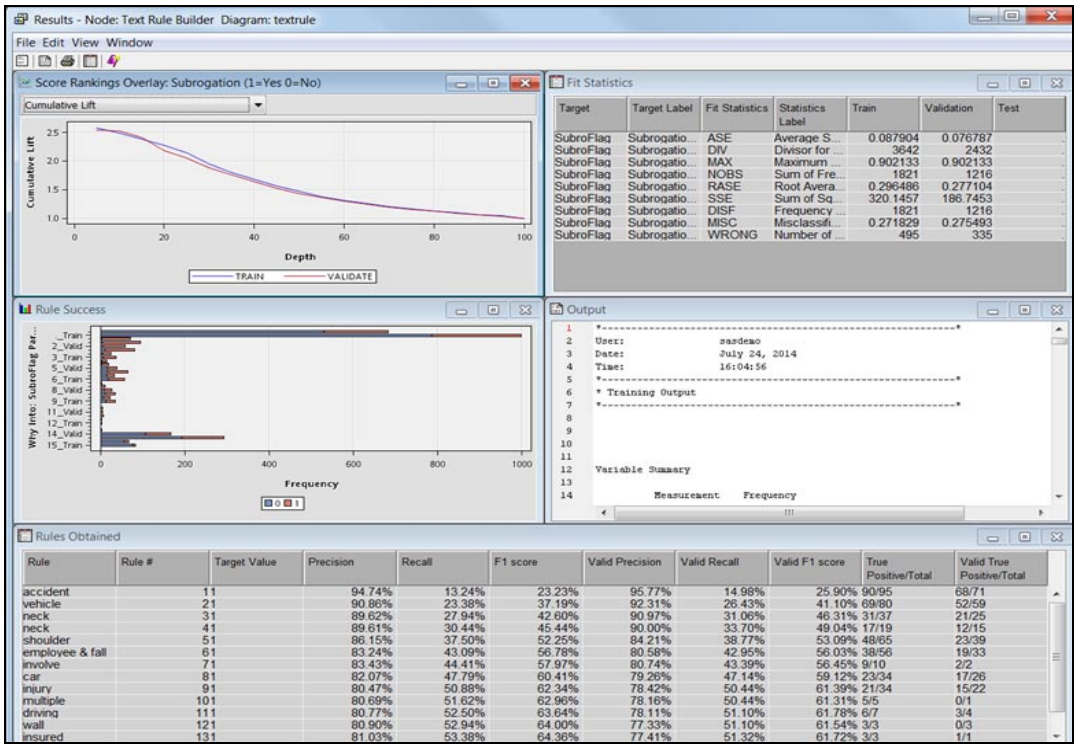
.. Property	Value
General	
Node ID	Part
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Output Type	Data
Partitioning Method	Default
Random Seed	12345
Data Set Allocations	
Training	60.0
Validation	40.0
Test	0.0
Report	
Interval Targets	Yes
Class Targets	Yes

4. Connect the following three text mining nodes (**Text Parsing**, **Text Filter**, and **Text Rule Builder**, in the same sequence) to the **Partition** node. With the default property settings, run the last node (**Text Rule Builder**) in the process flow.

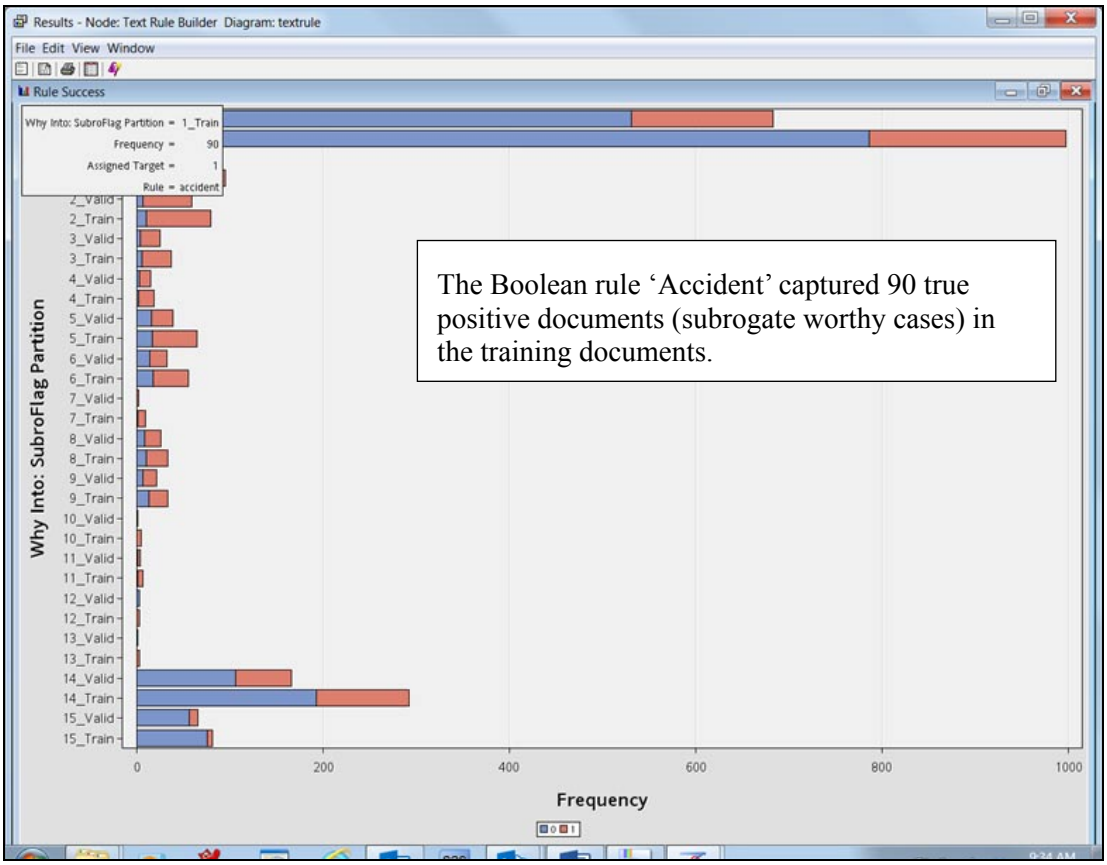


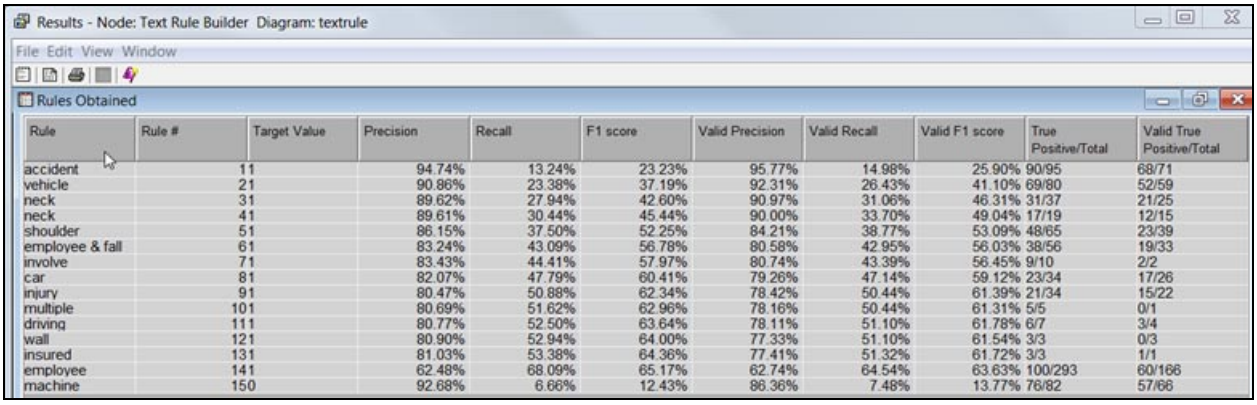
The Text Rule Builder node provides a stand-alone predictive modeling solution for data having a text variable and a categorical target variable. Thus, no Text Cluster or Text Topic node is required to generate predictions. However, Text Rule Builder must be preceded by Text Parsing and Text Filter nodes. The Text Rule Builder node creates Boolean rules from small subsets of terms to predict a categorical target variable. This specific target category consists of a conjunction that indicates the presence or absence of one or a small subset of terms (for example, **"term1" AND "term2" AND (NOT "term3")**). A particular document matches this rule if and only if it contains at least one occurrence of term1 and of term2 but no occurrences of term3.

5. View the results of the Text Rule Builder node.



The Rule Success chart and the Rule Obtained windows describe rule-based classification statistics for both training and validation data.





Rule	Rule #	Target Value	Precision	Recall	F1 score	Valid Precision	Valid Recall	Valid F1 score	True Positive/Total	Valid True Positive/Total
accident	11		94.74%	13.24%	23.23%	95.77%	14.98%	25.90%	90/95	68/71
vehicle	21		90.86%	23.38%	37.19%	92.31%	26.43%	41.10%	69/80	52/59
neck	31		89.62%	27.94%	42.60%	90.97%	31.06%	46.31%	31/37	21/25
neck	41		89.61%	30.44%	45.44%	90.00%	33.70%	49.04%	17/19	12/15
shoulder	51		86.15%	37.50%	52.25%	84.21%	38.77%	53.09%	48/65	23/39
employee & fall	61		83.24%	43.09%	56.78%	80.58%	42.95%	56.03%	38/56	19/33
involve	71		83.43%	44.41%	57.97%	80.74%	43.39%	56.45%	9/10	2/2
car	81		82.07%	47.79%	60.41%	79.26%	47.14%	59.12%	23/34	17/26
injury	91		80.47%	50.88%	62.34%	78.42%	50.44%	61.39%	21/34	15/22
multiple	101		80.69%	51.62%	62.96%	78.16%	50.44%	61.31%	5/5	0/1
driving	111		80.77%	52.50%	63.64%	78.11%	51.10%	61.78%	6/7	3/4
wall	121		80.90%	52.94%	64.00%	77.33%	51.10%	61.54%	3/3	0/3
insured	131		81.03%	53.38%	64.36%	77.41%	51.32%	61.72%	3/3	1/1
employee	141		62.48%	68.09%	65.17%	62.74%	64.54%	63.63%	100/293	60/166
machine	150		92.68%	6.66%	12.43%	86.36%	7.48%	13.77%	76/82	57/66

Based on the first 14 Boolean rules, 62.48% of all documents predicted as target=1 (Final precision %) and 68.09% of all documents where the observed target=1 (Final recall %) are considered true positives. Beyond the 14 rules used, no significant improvements in prediction was observed using F1 score criterion (validation harmonic mean of precision and recall %) at the default settings (Generalization Error=medium, Purity of Rules=medium, Exhaustiveness=medium)

The first Boolean rule used the term *accident*. Of the 95 documents containing the term *accident*, 90 have a **SubroFlag** target value of 1. The rule associated with row one is very simple: if the document contains the term *accident*, then classify the document as a subrogation (**SubroFlag**=1) document. Of the 95 documents so classified, 90 are correctly classified as a 1. Thus, the True Positive/Total column shows the value 90/95 for the accident rule in the training data. Of all of the rules considered, the accident rule provided the best rule for identifying recoveries. The sample precision is just the ratio 90/95 expressed as a decimal number.

Event Classification Table

Data Role=TRAIN Target=SubroFlag Target Label=Subrogation (1=Yes 0=No)

False Negative	True Negative	False Positive	True Positive
217	863	278	463

The training data contains 1821 (217+863+278+463) total, 680 observed positives (463+217), and 741 predicted positive (463+278) documents.

If only rule 1 were available, there would be a total of 95 claims classified as positive, leaving 1821-95=1726 claims classified as negative. Of these 1821 claims, 680-90=590 claims are false negatives, leaving 1821-680=1,141 true negatives. The confusion matrix for this first rule solution is given next.

	Classified as Positive	Classified as negative	Total
Actual Positive	90	590	680
Actual Negative	5	1136	1141
Total	95	1726	1821

The misclassification rate for this first rule solution is

MISC= (5+590)/1821=0.3267.

The precision and recall values are

PRECISION=90/95=0.947,

RECALL=90/680=0.1323,

which match the values supplied in the Rules Obtained table.

To help understand the subsequent rules after rule 1, consider the methodology used by the Text Rule Builder node. The node uses an algorithm analogous to the partitioning algorithm in decision tree construction. However, a decision tree partitions the data set based on the values of input variables, whereas the Text Rule Builder partitions based the contents of a document. Rules are usually simple, involving only a few words. The most complex rule in the above table uses only two words. The rule **employee & fall** classifies a document as a recovery if the document contains the word *employee* and the word *fall*. If a rule term includes the tilde (~), then the term acts as a negation operator. Just like a decision tree algorithm with binary splits, the Text Rule Builder systematically divides the document collection into two sets. One set satisfies the rule, leaving a second set of documents that do not satisfy the rule. The universe of rules to consider is massive, so part of the role of the algorithm is to limit the universe of rules, much like splitting options limit the number of splits that are considered by a decision tree. The options Generalization Error, Purity of Rules, and Exhaustiveness help limit the number of rules to consider.

The Generalization Error property is relevant when a validation data set is used. A higher generalization error does a better job of preventing overfitting than a lower value, but higher values also limit the rules to consider, possibly causing the algorithm to miss some important rules.

The Purity of Rules property controls the complexity of rules to consider. Selecting a high value produces "purer" rules than a low value. A rule with only a few words is considered to be purer than a rule with many words. Possible values are the same as the Generalization Error property.

The Exhaustiveness property determines how many rules will be considered. A high value permits many rules. Unlike the other two properties, a high value increases the time required to find useful rules. For the other two properties, high values decrease the time required to find useful rules. Possible values are the same as the Generalization Error property and the Purity of Rules property. If you are unsatisfied with the accuracy of the rules, then you could specify a lower value for either Generalization Error or Purity of Rules, or a higher value for Exhaustiveness. However, there is no guarantee that any combination of properties will always produce superior accuracy over any other combination of properties. Experimentation is usually required.

Consider the Rules Obtained window again. The simple accident rule is deemed to be the best rule, and all documents that satisfy the rule are removed from the training data. The algorithm is recursive, so it is applied to the new subset data, and another "best" rule is derived. In this instance, row two shows that the second best rule is *vehicle*. Of the 1821-95=1726 documents that remain after removing the 95 accident documents, 80 have the word *vehicle*. And of those, 69 have **SubroFlag**=1. If you divide 69 by 80, you get 0.8025, but the table shows a sample precision of 0. 0.9086. The sample precision for row two is the cumulative precision after applying the top two rules. The numerator is 90+69=159 true positives, out of 95+80=175 documents that satisfy the rules, leaving a sample precision of 0.9086. The confusion matrix for the sequential application of the first two rules follows.

	Classified as Positive	Classified as negative	Total
Actual Positive	159	521	680
Actual Negative	16	1125	1141
Total	175	1646	1821

The misclassification rate for the first two rule solution is

MISC=(16+521)/1821=0.295.

The cumulative precision and recall values for the first two rules are

PRECISION=159/175=0.9086,

RECALL=159/680=0.2338,

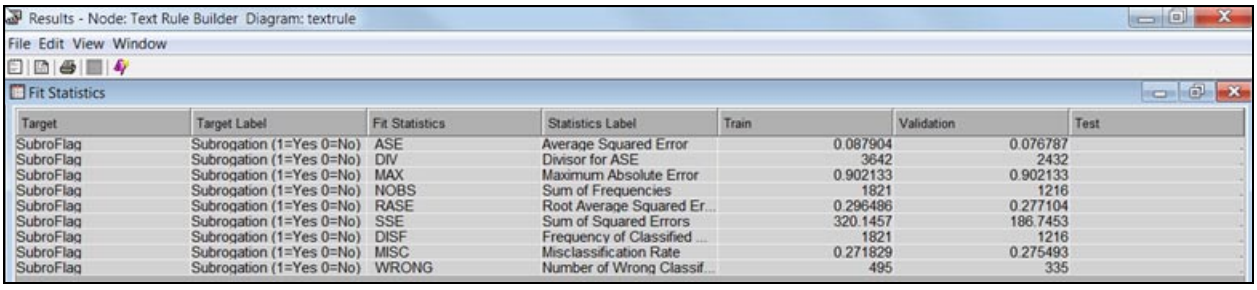
The rules are applied sequentially. If rule 1 is not satisfied, then rule 2 is investigated. If rule 2 is not satisfied, then rule 3 is investigated, and so on. This strategy is analogous to following the branches of a decision tree, except that when no rule is satisfied, the default action is to classify the document into the "failure" or "negative" class.

The Rules Obtained table contains 14 rules for **SubroFlag**=1 and 1 rule for **SubroFlag**=0. To understand the last one rules, you can note that for rule 15, before it was applied, there were 1,080 remaining documents, of which 217 have **SubroFlag**=1.

	Classified as Positive	Classified as negative	Total
Actual Positive	463	217	680
Actual Negative	278	863	1141
Total	741	1080	1821

Thus, when the algorithm gives up looking for positive rules and switches to negative rules step 15 (that is, rules for finding documents without recoveries), then it must search the 1,080 documents for the best negative rules. This explains how the last negative rules are found. When the rules switch from Target=1 to Target=0 for a binary response setting, the precision and recall statistics are recalibrated and only apply to the remaining data after all Target=1 rules have been applied.

The Fit Statistics table shows results from classifying all of the training and validation documents.


Results - Node: Text Rule Builder Diagram: textrule

File Edit View Window

Fit Statistics

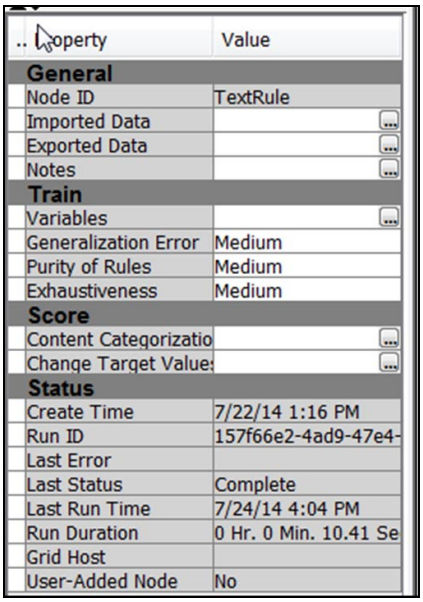
Target	Target Label	Fit Statistics	Statistics Label	Train	Validation	Test
SubroFlag	Subrogation (1=Yes 0=No)	ASE	Average Squared Error	0.087904	0.076787	
SubroFlag	Subrogation (1=Yes 0=No)	DIV	Divisor for ASE	3642	2432	
SubroFlag	Subrogation (1=Yes 0=No)	MAX	Maximum Absolute Error	0.902133	0.902133	
SubroFlag	Subrogation (1=Yes 0=No)	NOBS	Sum of Frequencies	1821	1216	
SubroFlag	Subrogation (1=Yes 0=No)	RASE	Root Average Squared Er...	0.296486	0.277104	
SubroFlag	Subrogation (1=Yes 0=No)	SSE	Sum of Squared Errors	320.1457	186.7453	
SubroFlag	Subrogation (1=Yes 0=No)	DISF	Frequency of Classified ...	1821	1216	
SubroFlag	Subrogation (1=Yes 0=No)	MISC	Misclassification Rate	0.271829	0.275493	
SubroFlag	Subrogation (1=Yes 0=No)	WRONG	Number of Wrong Classif...	495	335	

For example, the misclassification rate for the validation data is approximately 0.27. If you were examining a decision tree, you know that a decision tree produces a predicted probability for each observation. A cutoff value determines whether an observation is classified as a success or failure. You can change the cutoff and get different fit statistics. However, with the Text Rule Builder, there is no numeric cutoff. A rule is either satisfied or it is not. Thus, some fit statistics, like average squared error, do not seem to be relevant for scores coming from the Text Rule Builder because the Text Rule Builder only provides a zero-one score. However, you can use the sample precision for the validation data for a given rule partition to assign a numeric value between zero and one as a score, and then use this score in calculating fit statistics. Note that this is contrary to how scores in a decision tree are obtained. In general, empirical results from the training data are used as scores.

Active Learning By Changing Target Values

In the Text Rule Builder node, you have the option to change the target that is assigned (1 to 0) and rerun the results. Thus, the Text Rule Builder node facilitates *active learning* in which a user can dynamically interact with an algorithm to iteratively build a predictive model.

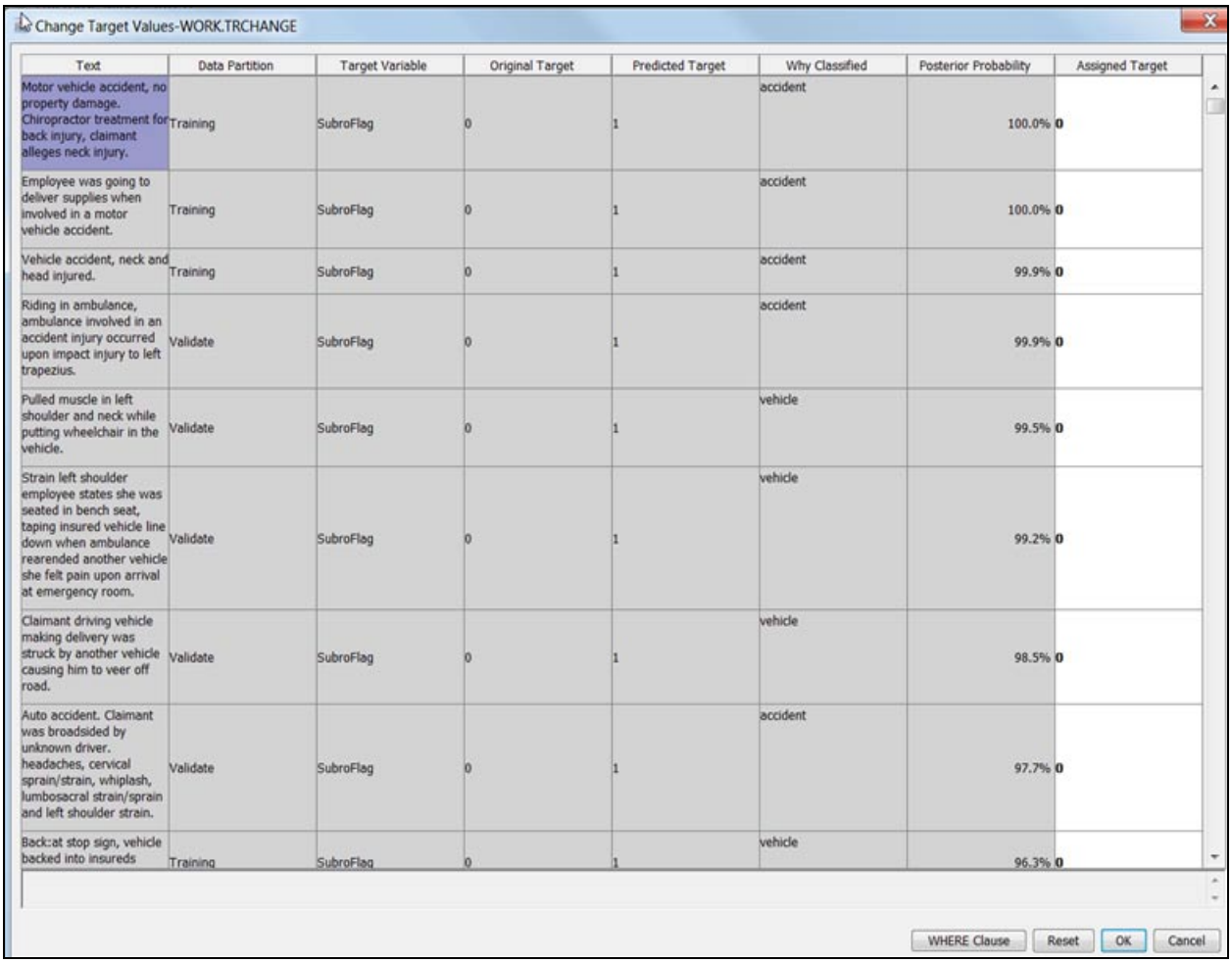
Click the ellipsis button to the right of the Change Target Values property to view the Change Target Values window. The Change Target Values window enables you to view and reassign target values.



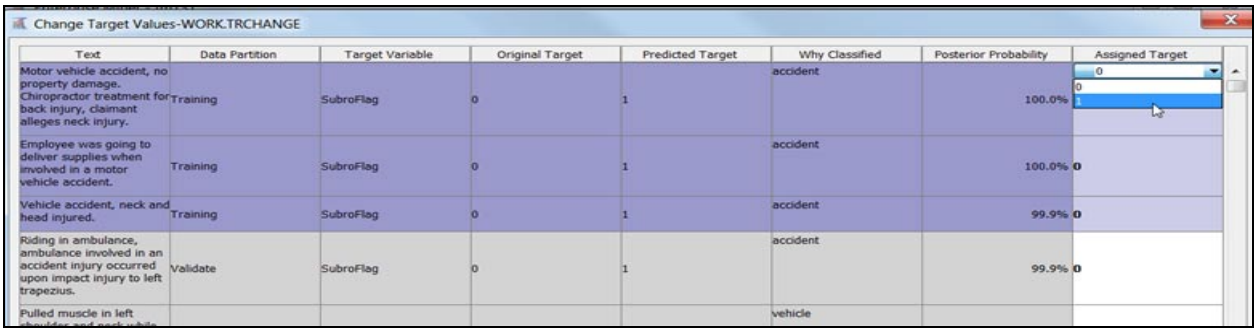
The observations in the Change Target Values window are ordered by the model's determined posterior probability in descending order from 1 to 0. Therefore, the values that the model is most certain are incorrect are at the very beginning.

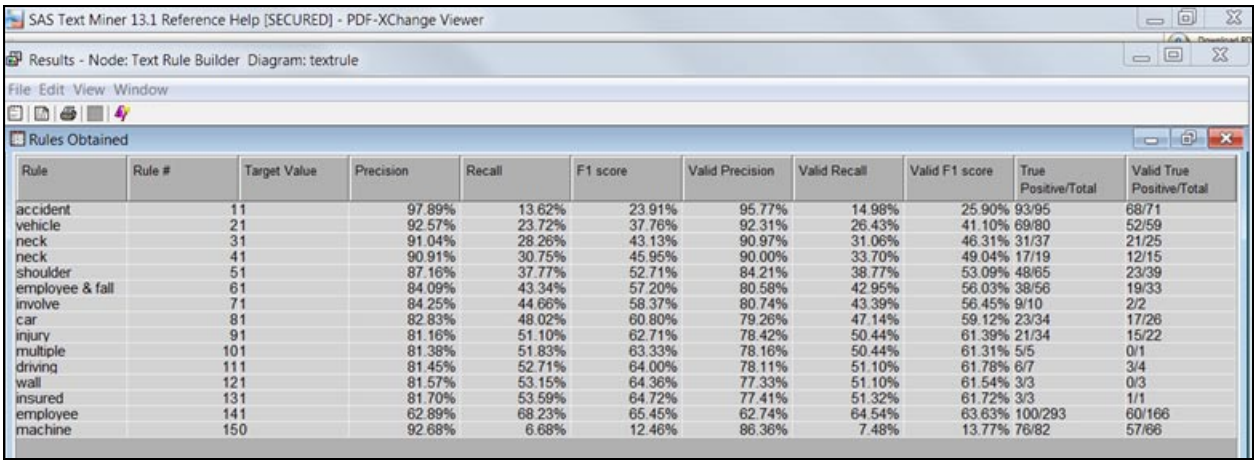
The data set that is shown in the Change Target Values window is not created until you run the node. The node will generate an error if you try to view the Change Target Values window before running the node. Any changes to the assigned target value are retained and used when the node is rerun, as long as the target variable has not been changed. When you rerun the node, your changes are applied to the data before the rule creation algorithm is run.

If you copy a Text Rule Builder node, then the Change Target Values data set is copied to the new node.



This window lists all the misclassify documents ranked by the posterior probability scores. The first three documents have almost 100% posterior probability associated with Target=1. These might be good candidates for reversing the assigned target value from 0 to 1. Then click to submit the changes and rerun the Text Rule Builder node.





Rule	Rule #	Target Value	Precision	Recall	F1 score	Valid Precision	Valid Recall	Valid F1 score	True Positive/Total	Valid True Positive/Total
accident	11		97.89%	13.62%	23.91%	95.77%	14.98%	25.90%	93/95	68/71
vehicle	21		92.57%	23.72%	37.76%	92.31%	26.43%	41.10%	69/80	52/59
neck	31		91.04%	28.26%	43.13%	90.97%	31.06%	46.31%	31/37	21/25
neck	41		90.91%	30.75%	45.95%	90.00%	33.70%	49.04%	17/19	12/15
shoulder	51		87.16%	37.77%	52.71%	84.21%	38.77%	53.09%	48/65	23/39
employee & fall	61		84.09%	43.34%	57.20%	80.58%	42.95%	56.03%	38/56	19/33
involve	71		84.25%	44.66%	58.37%	80.74%	43.39%	56.45%	9/10	2/2
car	81		82.83%	48.02%	60.80%	79.26%	47.14%	59.12%	23/34	17/26
injury	91		81.16%	51.10%	62.71%	78.42%	50.44%	61.39%	21/34	15/22
multiple	101		81.38%	51.83%	63.33%	78.16%	50.44%	61.31%	5/5	0/1
driving	111		81.45%	52.71%	64.00%	78.11%	51.10%	61.78%	6/7	3/4
wall	121		81.57%	53.15%	64.36%	77.33%	51.10%	61.54%	3/3	0/3
insured	131		81.70%	53.59%	64.72%	77.41%	51.32%	61.72%	3/3	1/1
employee	141		62.89%	68.23%	65.45%	62.74%	64.54%	63.63%	100/293	60/166
machine	150		92.68%	6.68%	12.46%	86.36%	7.48%	13.77%	76/82	57/66

Examine the revised Precision and Recall percentages. The number of true positive frequencies has increased for both *accident* and the precision rule.

A **Model Comparison** node can be connected to **Text Rule Builder** node to assess the performance of the node with other EM predictive modeling nodes. The Chapter 2 demonstration of this data considers a variety of predictive models. Validation average squared error is the fit criterion that is used. Because this does not seem appropriate for this Text Rule Builder node, validation misclassification percentage could be used.

4.3 High Performance (HP) Text Miner Node

Objectives

- Identify the benefits of the HP Text Miner node.
- Run a process flow using HP components.
- State the capabilities within the HPTMINE procedure.

43

High-Performance Text Mining

The ***HPTMINE*** procedure and node is designed to

- process extremely large amounts of text data
- do so quickly.

The HP environment requires a special implementation of Enterprise Miner:

- grid enablement
- SMP (symmetric multiprocessing – multi-threads)
- MPP (massively parallel processing)

44

You must use a *high-performance configuration* to effectively take advantage of this capability.

HP Modes of Operation

Distributed Mode

- Several configurations
- "Alongside the database"

Single-Machine Mode

- With local data
- Does not use the MPP environment
  - (MPP = Massively parallel processing)

45

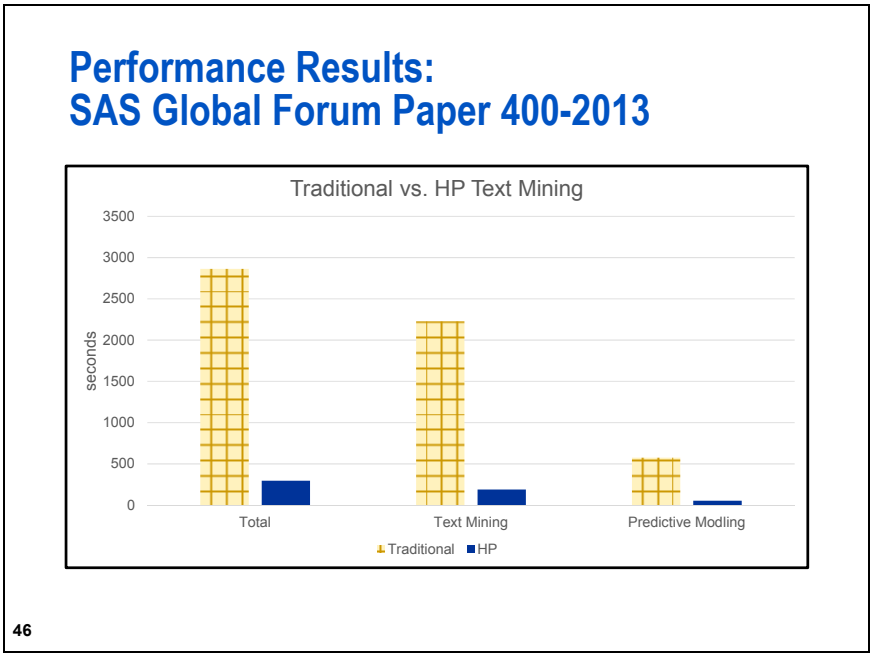
Distributed mode is the more typical deployment of an HP configuration where significant reductions in elapsed time and improvements in efficiency can be achieved. The high-performance procedure runs on several computers that are called an *appliance*. The appliance can include a database management system. "Alongside the database" mode supports parallel reading of data into a high-performance analytical procedure running on the database appliance.

In single-machine mode, a single computer operating system oversees the running of the HP processes. Processors, disks, and memory in this mode can be used by the analytical processes. The multithreading capabilities built in to Enterprise Miner work in this mode. It is possible to combine some HP nodes with non-HP nodes successfully in an Enterprise Miner process flow, although this is not considered a best practice.

For additional details refer to

The book: *Base SAS® 9.4 Procedures Guide: High-Performance Procedures, Second Edition*

The course: SAS® Enterprise Miner High-Performance Data Mining Nodes



The results above were presented in the SAS Global Forum 2013 paper 400-2013.
http://support.sas.com/resources/papers/proceedings13/400-2013.pdf

The data consisted of more than 680,000 paragraphs from the Consumer Complaints data set. The traditional elapsed times were run on a Windows server with two CPUs and 128 GB of memory. The HP nodes were run on a cluster containing 16 computing nodes, each with two CPUs and 64 GB of memory. The high- performance text mining procedures can reduce a 30-minute task to less than a minute in a grid computing environment according to findings presented in this paper.

High-performance results can vary and depend on the configuration of your SAS environment and your specific parallel processing machine configuration.

High-Performance Nodes

The HP Text Miner node (HPTM) is one of many tools available on the SEMMA tab shown here (HPDM). HP Text Miner executes two phases:

- text parsing
- transformation

47

HPTM Properties

Property	Value
General	
Node ID	HPTM
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Detect	
Different Parts of Speech	Yes
Find Entities	No
Multi-word Terms	SASHELP.ENG_MULTI
Synonyms	SASHELP.ENGSYNMS
Filter	
Stop List	SASHELP.ENGSTOP
Minimum Number of Documents	4
Transform	
SVD Resolution	Low
Max SVD Dimensions	100
Report	
Number of Terms to Display	20000
Status	
Create Time	6/6/14 1:41 PM

48

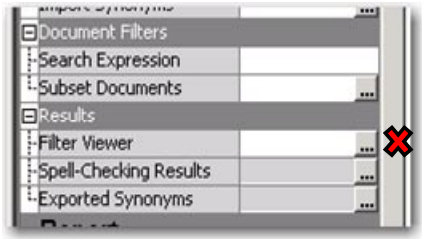
The HP Text Miner node property selections include Detect, Filter, and Transform. Text parsing, natural language processing, and entity detection are all supported.

Your own pre-existing customized tables can be specified for multi-word terms, synonyms, and stop lists just as in the regular Text Mining nodes. The SVD (Singular Value Decomposition) resolution can be any number from 2 to 500. A higher number generates a better data summary but takes more computing power to finish.

The Max SVD Dimensions property (maxdim) accounts for p% of the total variance. High resolution always generates the maximum number of SVD dimensions (maxdim). For medium resolution, the recommended number of SVD dimensions accounts for 5/6*(p% of the total variance). For low resolution, the recommended number of SVD dimensions accounts for 2/3*(p% of the total variance).

Compare Text Mining Node Properties

Unlike the Text Filter node, the HP node does not have a Filter Viewer ellipsis.



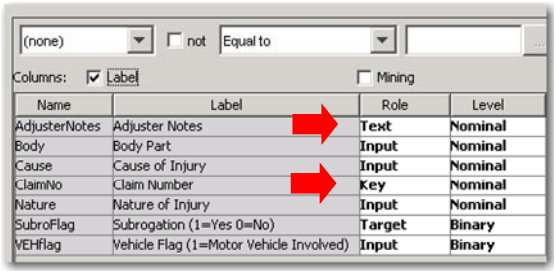
49

The property shown here is from the regular Text Filter node. This option was not made available in the High-Performance Text Miner node.

Data Set Requirements

The HP Text Miner node requires an input data set that contains the following

- a variable with the role **Text**. This variable cannot be an interval variable
- a variable with the role **Key**. The key variable contains a unique identifier for each observation in the input data set (similar to the role ID)



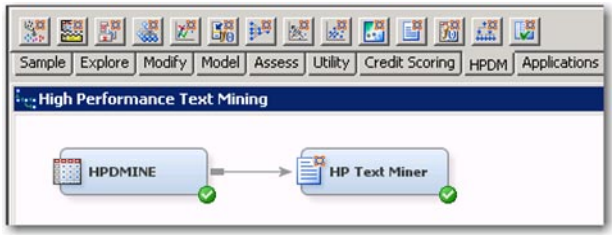
50

The role KEY must be set when the data source is defined to Enterprise Miner. It is not available in the Metadata node. There are seven variables in this data set, including four inputs, a text, a key, and a target. A target variable is not necessary for unsupervised text mining.

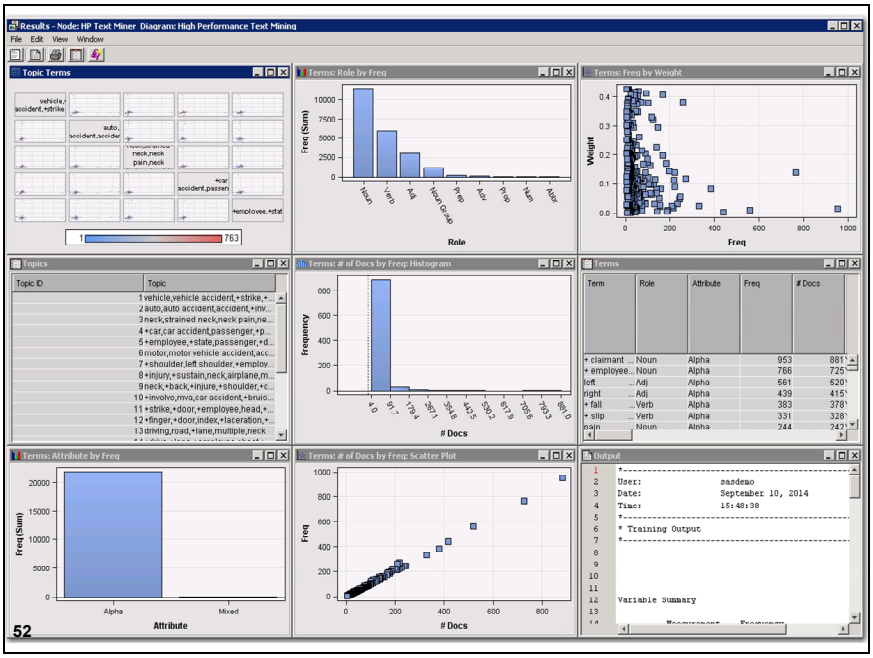
Process Flow

A data source is connected to the HP Text Miner node in this diagram. It has been run, and the results are on the next slide.

All properties in this example were left as their defaults.



51



52

The Topics and Terms table results are contained in the node results. The maximum number of terms shown in the table is 20,000 by default. The other plots in this window provide the same insight into the document collection as described in previous chapters.

Output Window

The Output window lists the tasks that were run. The HPTMINE procedure consolidates the actions that would be performed by several non-high-performance nodes.

```
24  The HPTMINE Procedure
25
26        Performance Information
27
28  Execution Mode       Single-Machine
29  Number of Threads    2
30
31
32                  Procedure Task Timing
33
34  Task                                  Seconds     Percent
35
36   Parse documents                         3.22      86.41%
37   Analyze terms                           0.05       1.42%
38   Obtain term frequency                   0.09       2.38%
39   Filter terms                            0.03       0.70%
40   Generate term-document matrix           0.00       0.08%
41   Output OUTTERMS table                   0.00       0.05%
42   Output OUTCONFIG table                  0.00       0.03%
43   Compute SVD                             0.33       8.84%
44   Output SVDU table                       0.00       0.08%
45
```

53

Topics

The SVD process was used to derive the list of topics from the document collection. This list used the default Low resolution setting.

Topics

Topic ID	Topic
20	repetitive,+motion,+repetitive motion,due to,tendonitis
21	lot,+park,+park lot,ice,+slip
22	+patient,ambulance,+claimant,+step,head
23	right,+knee,+shoulder,+hand,+arm
24	+year,old,yr,+allege,old
25	+employee,+state,+strain,+lift,back
26	+wrist,left wrist,+sprain,left,strained
27	+leg,left leg,+cause,left,dock
28	+thumb,left thumb,left,+claimant,+unload
29	+pull,+muscle,+slip,groin,back
30	ladder,+fall,balance,lost,+employee

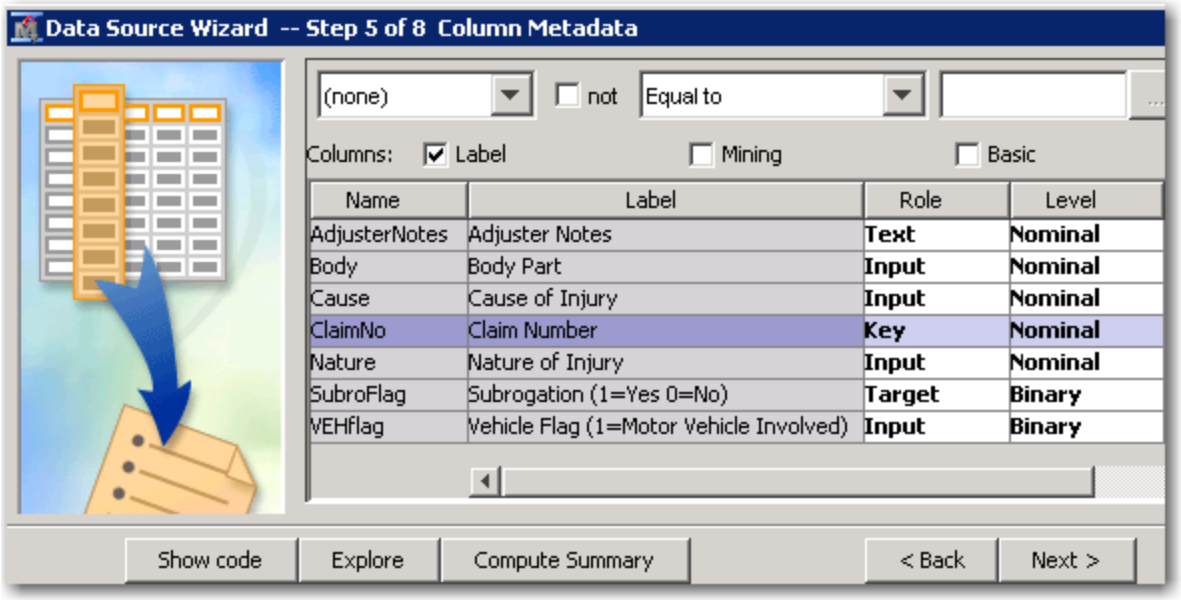
54

Predictive Modeling with the HP Text Miner Node

This demonstration illustrates the capabilities and results of the High-Performance Text Miner node in a single-machine environment.

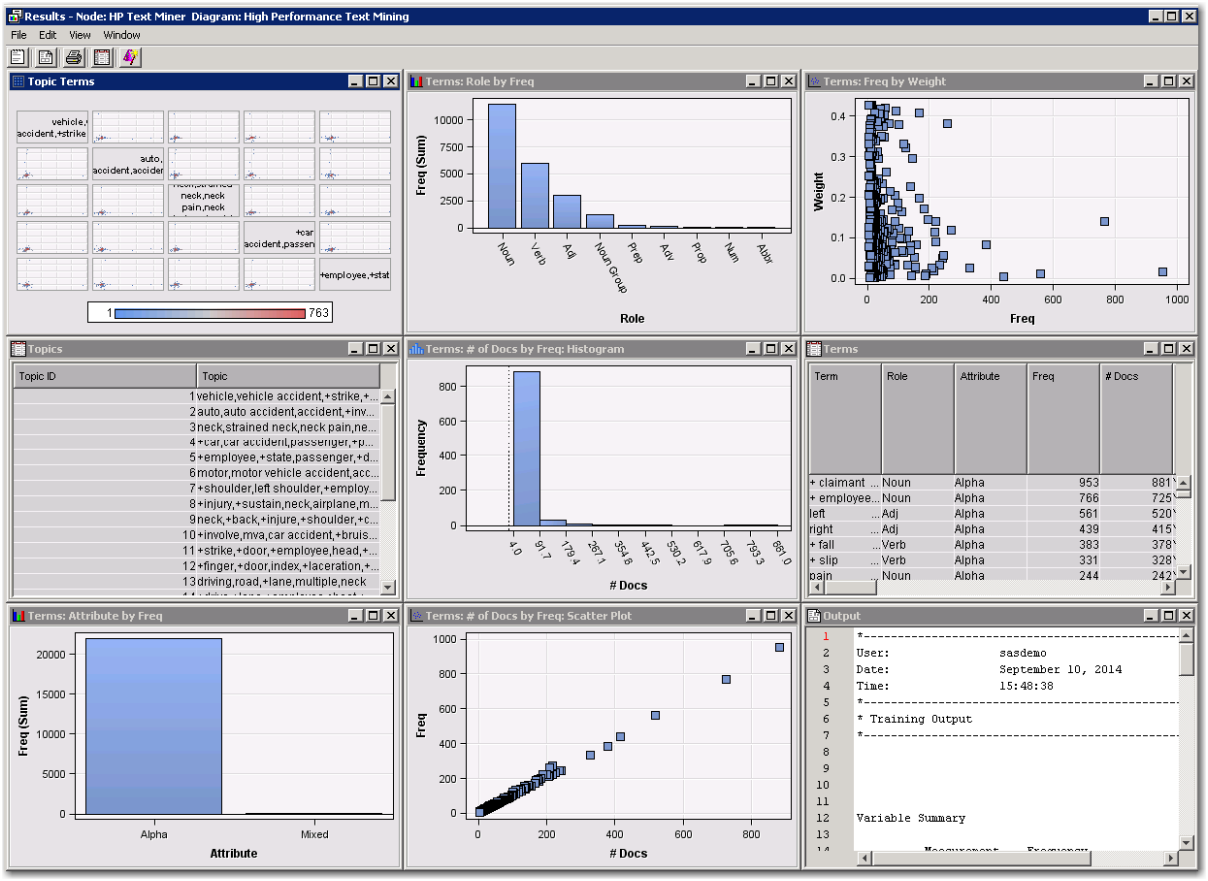
1. Create a new diagram named **High-Performance Text Mining** in Enterprise Miner.

2. Create a new data source from the **DMTXT** library using the **HPDMINE** table. Make sure that you set the role to **Key** for the **ClaimNo** variable. Set the Role and Level attributes for each variable according to the display capture below. Pull the data source into the diagram.



3. From the HPDM SEMMA tab, pull an **HP Text Miner** node into the diagram and connect the data source to it.

4. Run the node with all the default settings, and open the results.



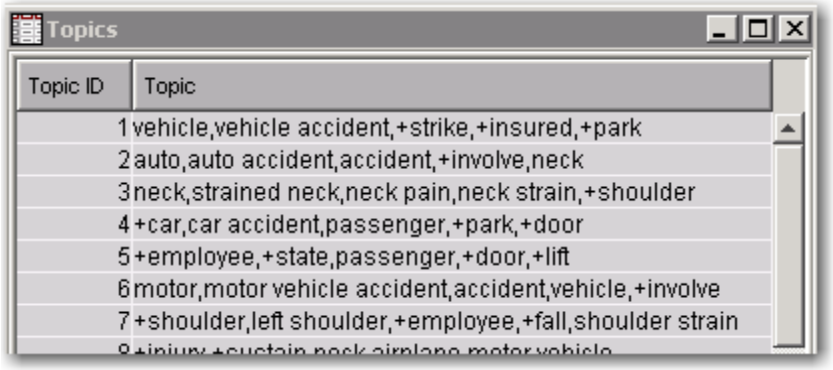
5. Maximize the Output window. The results indicate that the HPTMINE procedure ran in single-machine mode. The mode depends on the type of Enterprise Miner implementation. The Output window shows that the document collection was parsed, terms were analyzed and filtered, and singular value decomposition was done.

```
24    The HPTMINE Procedure
25
26          Performance Information
27
28    Execution Mode       Single-Machine
29    Number of Threads    2
30
31
32                   Procedure Task Timing
33
34    Task                                    Seconds     Percent
35
36     Parse documents                           3.22      86.41%
37     Analyze terms                             0.05       1.42%
38     Obtain term frequency                     0.09       2.38%
39     Filter terms                              0.03       0.70%
40     Generate term-document matrix             0.00       0.08%
41     Output OUTTERMS table                     0.00       0.05%
42     Output OUTCONFIG table                    0.00       0.03%
43     Compute SVD                               0.33       8.84%
44     Output SVDU table                         0.00       0.08%
45
```

6. Examine the fields of the Terms and Topics tables that were created. Note how similar (and familiar) they are compared to the Text Mining nodes that we ran in previous chapters.

Terms

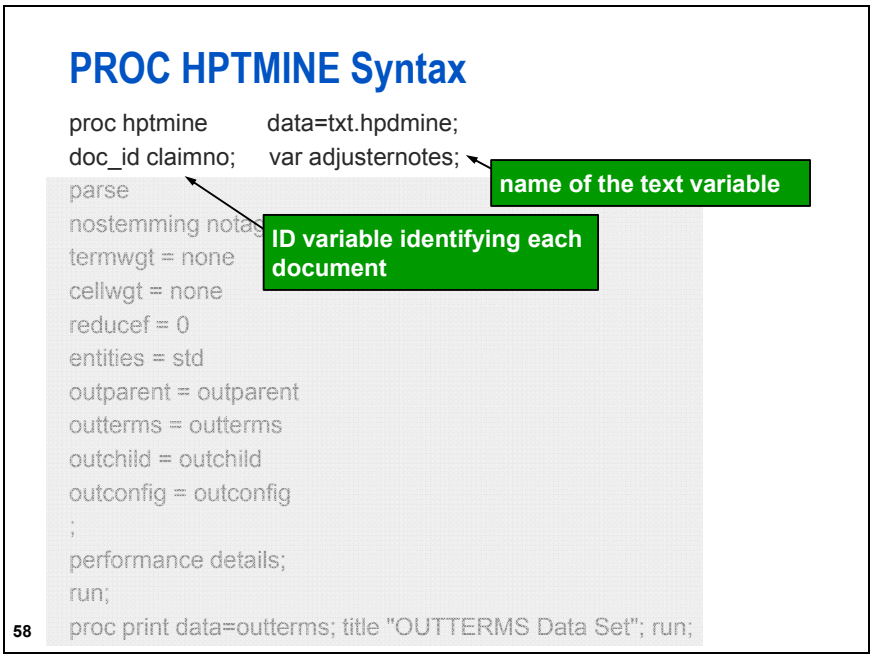
Term	Role	Attribute	Freq	# Docs	Keep	Weight	Rank for Variable numdocs	vehicle,vehicle accident,+strike,+insured,+park	auto,auto accident,accident,+involve,neck
+ claimant	Noun	Alpha	953	881	Y	0.015	1	0.005	0.001
+ employee	Noun	Alpha	766	725	Y	0.139	2	0.048	0.044
left	Adj	Alpha	561	520	Y	0.011	3	0.003	0.002
right	Adj	Alpha	439	415	Y	0.003	4	0	0
+ fall	Verb	Alpha	383	378	Y	0.083	5	-0.001	-0.005
+ slip	Verb	Alpha	331	328	Y	0.026	6	0.001	-0.001
pain	Noun	Alpha	244	242	Y	0.048	7	0.001	-0.003



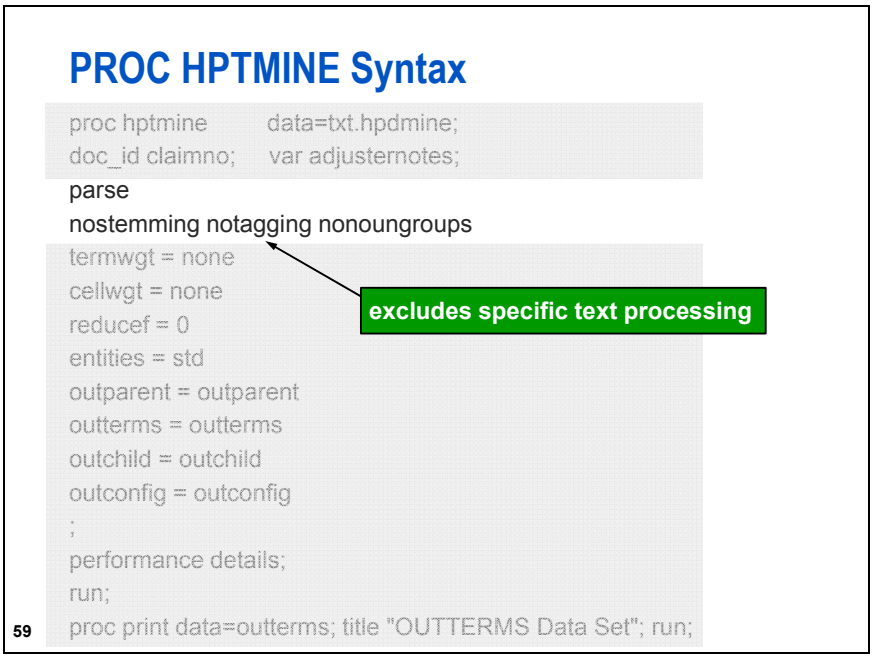
PROC HPTMINE Syntax (Optional)

PROC HPTMINE can be coded to run in batch mode. The following slides illustrate the syntax that will be used in the next demonstration.

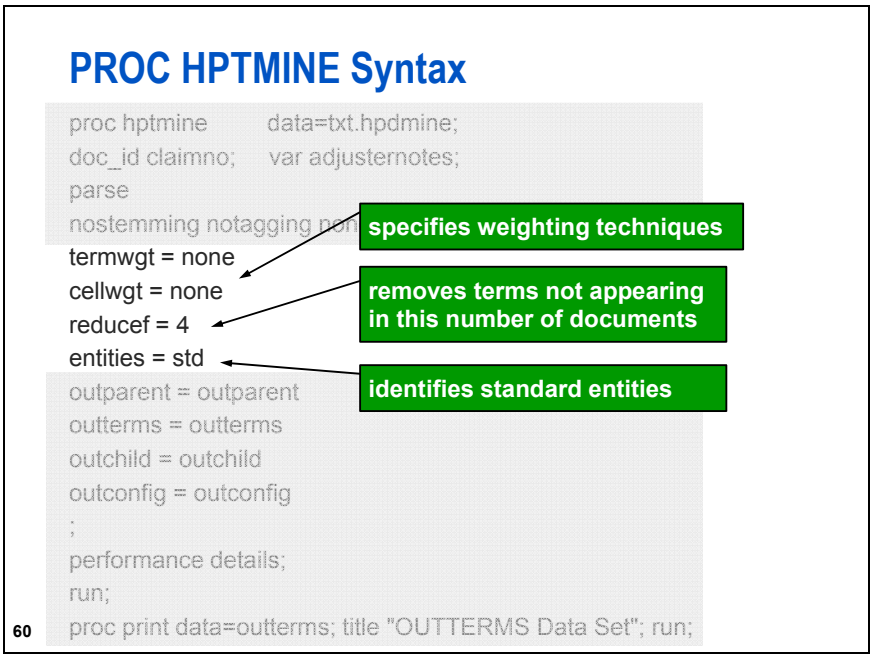
57



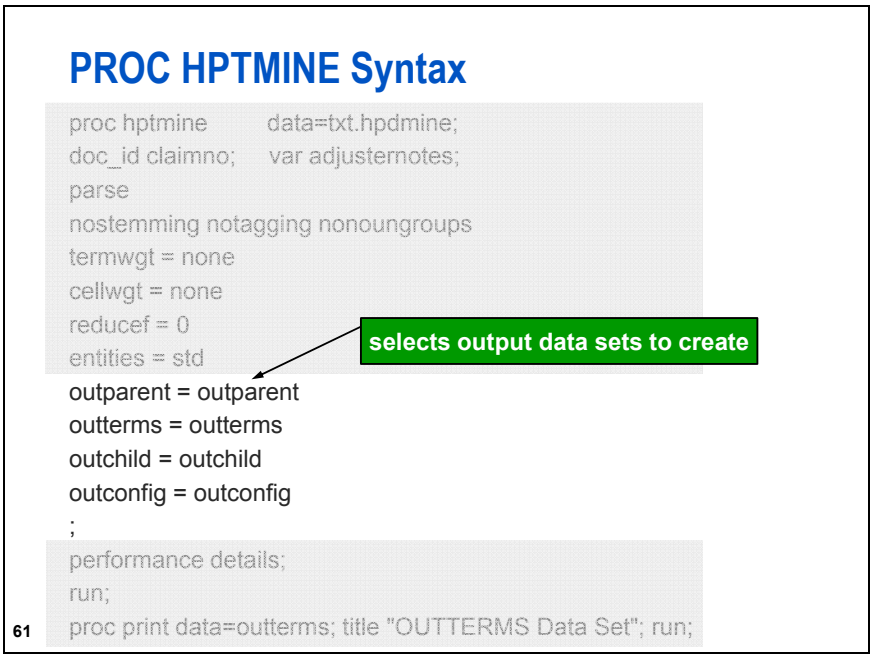
Declare the data set, ID (or key) variable, and the name of the text variable. The data set you use in batch mode will likely not have the Data Mining metadata attributes, so you have to identify these variables in the procedure as specified.



These lines state that we want to parse the document collection. Stemming, tagging, and noun grouping operations can be performed or suppressed by excluding or including these variables.



Term and cell weights are applied to the compressed term-by-document matrix. Term weight can be Entropy, MI, or None. Cell weight can be Log or None. Terms appearing in fewer than the reduced number of documents will be excluded from analysis. Entities can either be identified or ignored.



Term number, document, and count variables will be produced in these output data sets. The OUTCONFIG= data set is used if a subsequent HPTMSCORE procedure will be run against the document collection results.

PROC HPTMINE Syntax

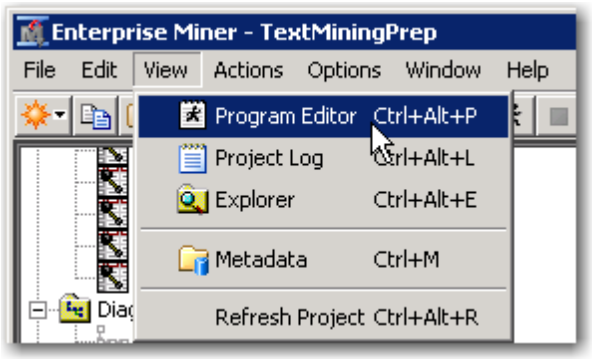
```
proc hptmine        data=txt.hpdmine;
doc_id claimno;     var adjusternotes;
parse
nostemming notagging nonoungroups
termwgt = none
cellwgt = none
reducef = 0
entities = std
outparent = outparent
outterms = outterms
outchild = outchild
outconfig = outconfig
;
performance details;
run;
62  proc print data=outterms; title "OUTTERMS Data Set"; run;
```

shows elapsed time of tasks

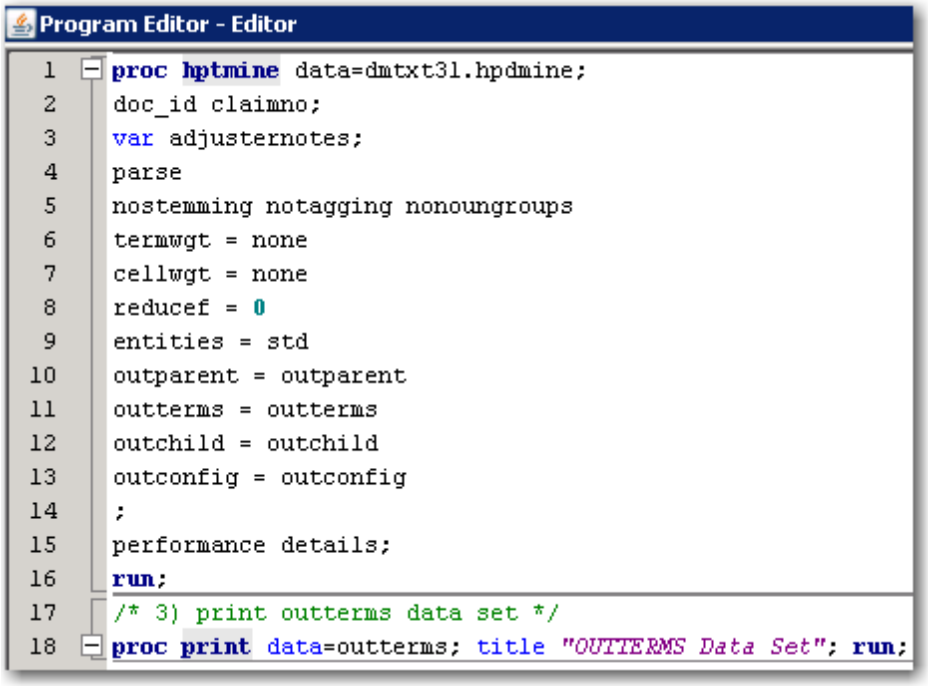
Using PROC HPTMINE

This demonstration illustrates use of the HPTMINE procedure.

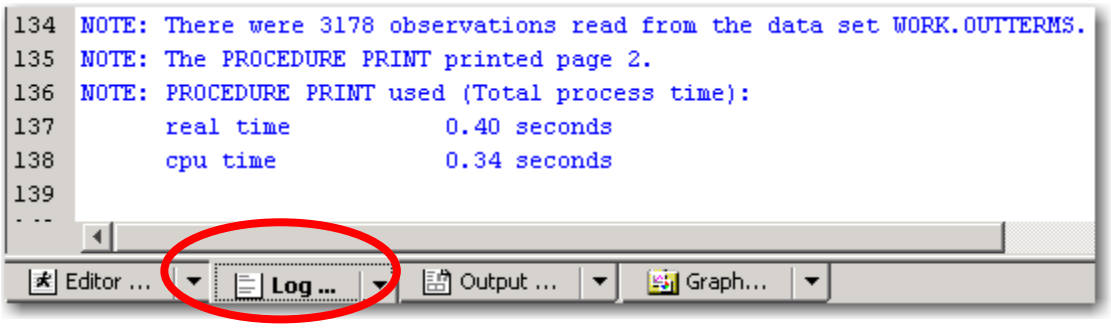
1. From Enterprise Miner, select **View** ⇨ **Program Editor**.



2. From the Editor, select **File ⇨ Open ⇨ d:\workshop\winsas\dmtxt31\sassrc\hptmine.sas**. The following program appears:



3. Submit the code by pressing F3 or clicking the **Run** icon. Check the log.



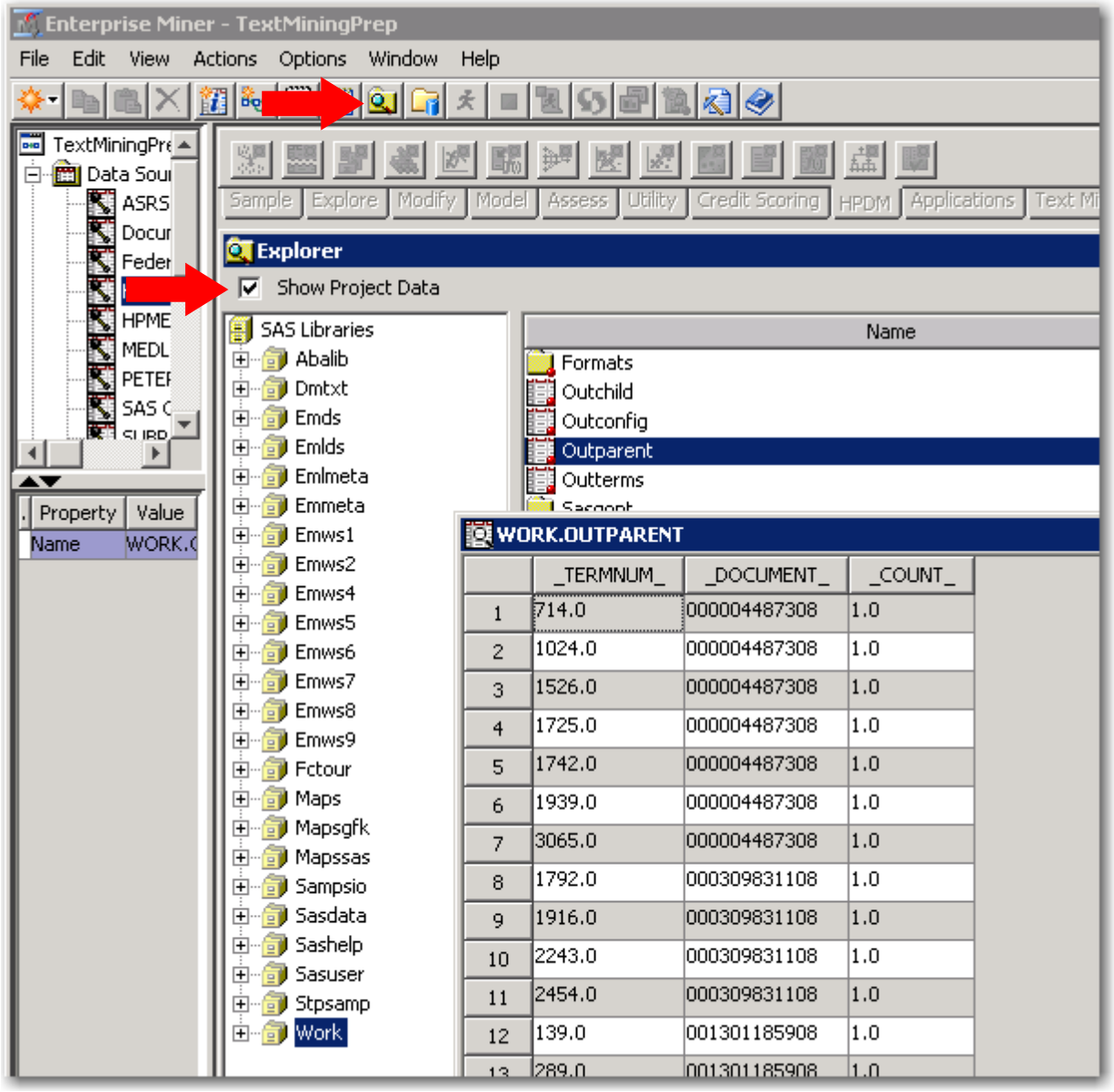
Open the Output window.

4. Observe the contents of the Output window to see the task timing and the OUTTERMS= data set results.

Program Editor - Output

```
1
2  1
3
4  The HPTMINE Procedure
5
6        Performance Information
7
8  Execution Mode       Single-Machine
9  Number of Threads    2
10
11
12                 Procedure Task Timing
13
14 Task                              Seconds     Percent
15
16  Parse documents                     5.20     96.17%
17  Analyze terms                       0.01      0.26%
18  Obtain term frequency               0.15      2.83%
19  Filter terms                        0.01      0.20%
20  Generate term-document matrix       0.01      0.19%
21  Output OUTPARENT table              0.01      0.13%
22  Output OUTCHILD table               0.01      0.11%
23  Output OUTTERMS table               0.00      0.09%
24  Output OUTCONFIG table              0.00      0.02%
25 OUTTERMS Data Set
26 2
27
28                                                                                  Parent_
29  Obs    Term                     Role          Attribute    Freq    numdocs    _keep    Key    Parent      id
30
31    1    felt                                   Alpha          98         98    Y          1         .       1
32    2    recv                                   Alpha           1          1    Y          2         .       2
33    3    racks                                  Alpha           8          8    Y          3         .       3
34    4    blowout                                Alpha           1          1    Y          4         .       4
35    5    unclear                                Alpha           1          1    Y          5         .       5
36    6    program director         TITLE         Entity          1          1    Y          6         .       6
37    7    up                                     Alpha         142        138    Y          7         .       7
```

5. (Optional) Open SAS Explorer, select the **Show Project Data** box, and select the **Work** library. You will be able to see the output data sets created by the HPTMINE procedure. The OUTPARENT= data set is shown below.

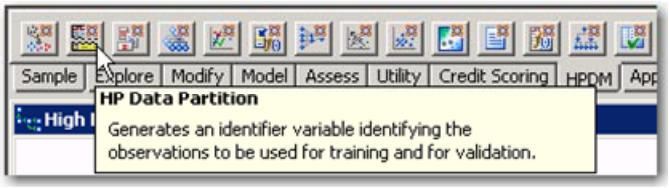


	TERMNUM	_DOCUMENT_	_COUNT_
1	714.0	000004487308	1.0
2	1024.0	000004487308	1.0
3	1526.0	000004487308	1.0
4	1725.0	000004487308	1.0
5	1742.0	000004487308	1.0
6	1939.0	000004487308	1.0
7	3065.0	000004487308	1.0
8	1792.0	000309831108	1.0
9	1916.0	000309831108	1.0
10	2243.0	000309831108	1.0
11	2454.0	000309831108	1.0
12	139.0	001301185908	1.0
13	289.0	001301185908	1.0

HP Data Partition Node

If your objective is prediction in a high-performance configuration, you must use the HP Data Partition node from the HPDM tab for compatibility.

Training and Validation partitions are supported. A Test partition is not available in this mode.



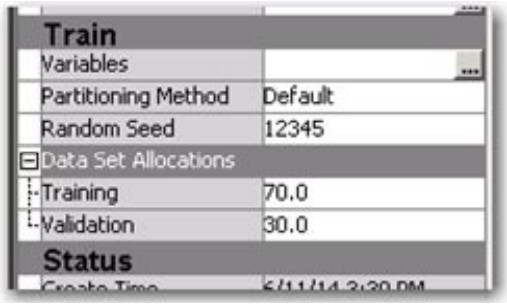
64

If you are running in single-machine mode as in the classroom environment, the original Data Partition node could be used, and will not generate any errors. You could use either Data Partition node in this specific case.

However, it is a best practice to use the HP Data Partition node with the HP nodes because this is necessary in distributed mode.

HP Partition Properties

These are the selections available in the HP partition node:

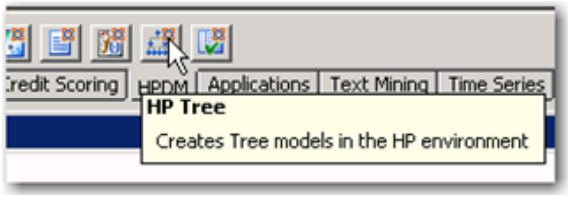


65

The HP Data Partition node supports two types of partitioning: simple random and stratified. With a class target variable, the default method is stratified partitioning. Otherwise, simple random partitioning is used. The node supports up to two stratification variables.

HP Tree

There is a high-performance decision tree node that runs in a high-performance environment. Find it on the HPDM tab of the SEMMA palette.



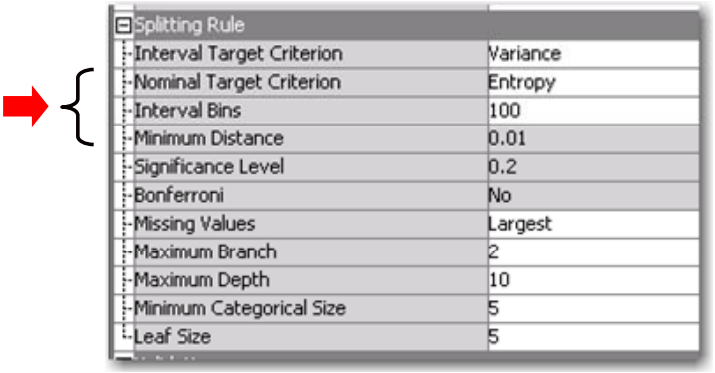
66

Although additional high-performance modeling nodes are available, we will look at only the HP Tree node. Explore the other modeling nodes and compare the results to select your champion model.

HP Tree Property

A few new properties are in the HP Tree node.

- Nominal Target Criterion Fast Chaid
- Interval Bins – for interval variables
- Minimum KS distance for Fast Chaid trees



Splitting Rule	
Interval Target Criterion	Variance
Nominal Target Criterion	Entropy
Interval Bins	100
Minimum Distance	0.01
Significance Level	0.2
Bonferroni	No
Missing Values	Largest
Maximum Branch	2
Maximum Depth	10
Minimum Categorical Size	5
Leaf Size	5

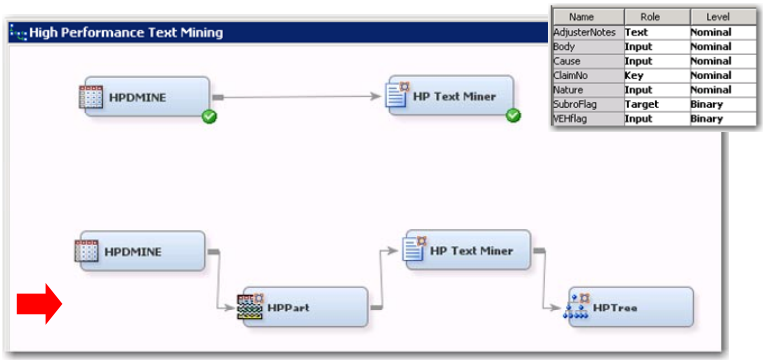
67

There is also a property to create a validation sample from the training data set. If you have a validation data set available, it could be used in lieu of a sample.

Notable properties that are unavailable with this node include the Interactive tree viewer, Decisions and Priors, and Cross Validation.

Process Flow

In this demonstration, copy the previous process flow and add the HP Partition and HP Tree nodes with the default properties. The copied and modified flow is shown below. The data source has a binary target variable: **SubroFlag**.



68

Results

The fit statistics from the results of running the HPTree node with default settings are below. This run of the node used the Validation partition from the HPPart node because we did not request that it create its own validation set.

Fit Statistics	Statistics Label	Train	Validation
ASE	Average Squared Error	0.147365	0.169427
DIV	Divisor for ASE	4252	1822
MAX	Maximum Absolute Error	0.96	1
NOBS	Sum of Frequencies	2126	911
RASE	Root Average Squared Error	0.383882	0.411615
SSE	Sum of Squared Errors	626.5971	308.696
DISF	Frequency of Classified Cases	2126	911
MISC	Misclassification Rate	0.209784	0.226125
WRONG	Number of Wrong Classifications	446	206

69

Predictive Modeling Using High-Performance Nodes

This demonstration illustrates how to perform predictive modeling using high-performance nodes.

1. Open the **High-Performance Text Mining** diagram from the previous demonstration.
2. Copy the two nodes in the diagram (**HPDMINE** and **HP Text Miner**), and paste them in the diagram below the originals.

3. Drag an **HP Data Partition** node and an **HP Tree** node into the diagram from the HPDM tab.

4. Connect the nodes in the order shown below. Run the process flow from the end with default settings.

5. Close the completion window.

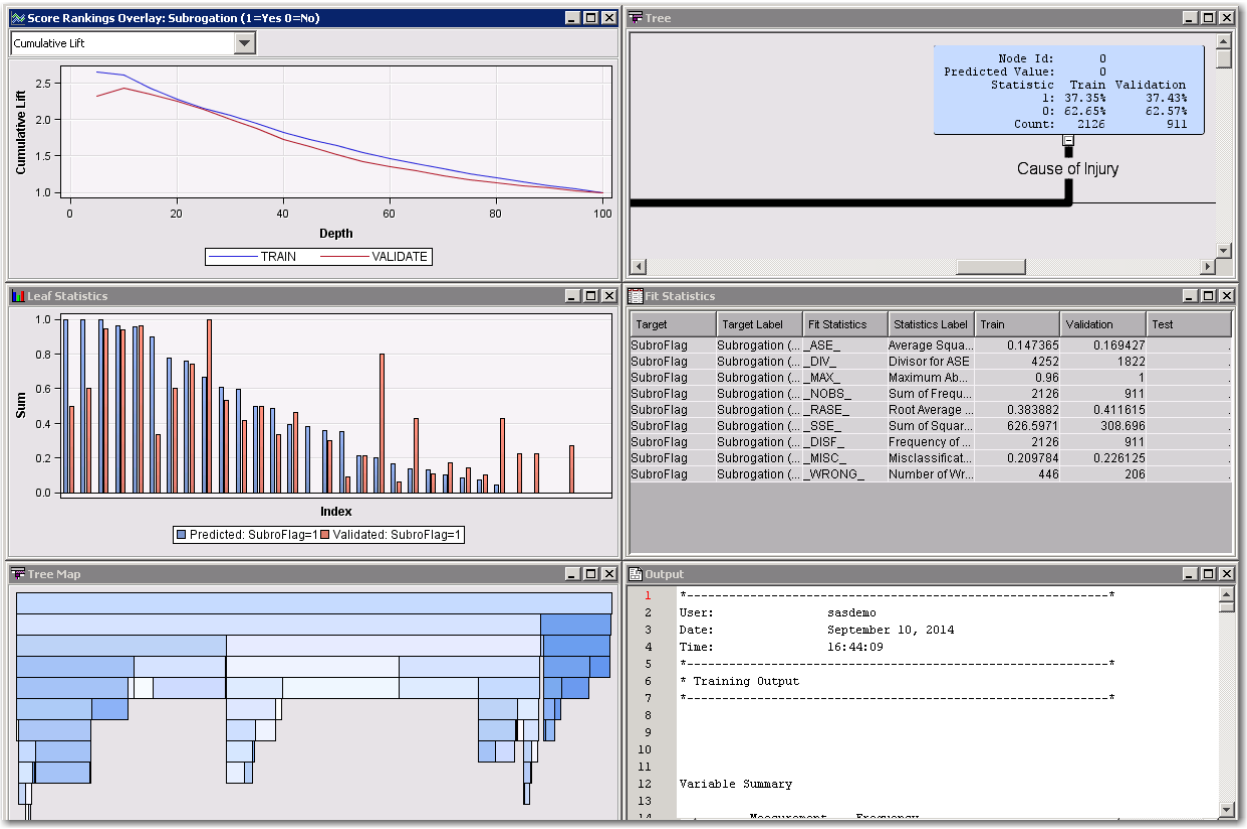
6. Examine the properties and the results of the HPPart node and verify that 70% of the data was allocated to the role: Train. (341 / 1135 = .30)

Partition Information

Stratification Variable	Number of observations	Training Observations	Validation Observations
0	1902	1332	570
1	1135	794	341

7. Look at the results of the HP Text Miner node. Notice that fewer topics were derived compared to the previous run.

8. Open the results of the HP Tree node. Are there any property settings that you would consider testing to possibly create an even better predictive model? Look at the Leaf Statistics node for a hint.

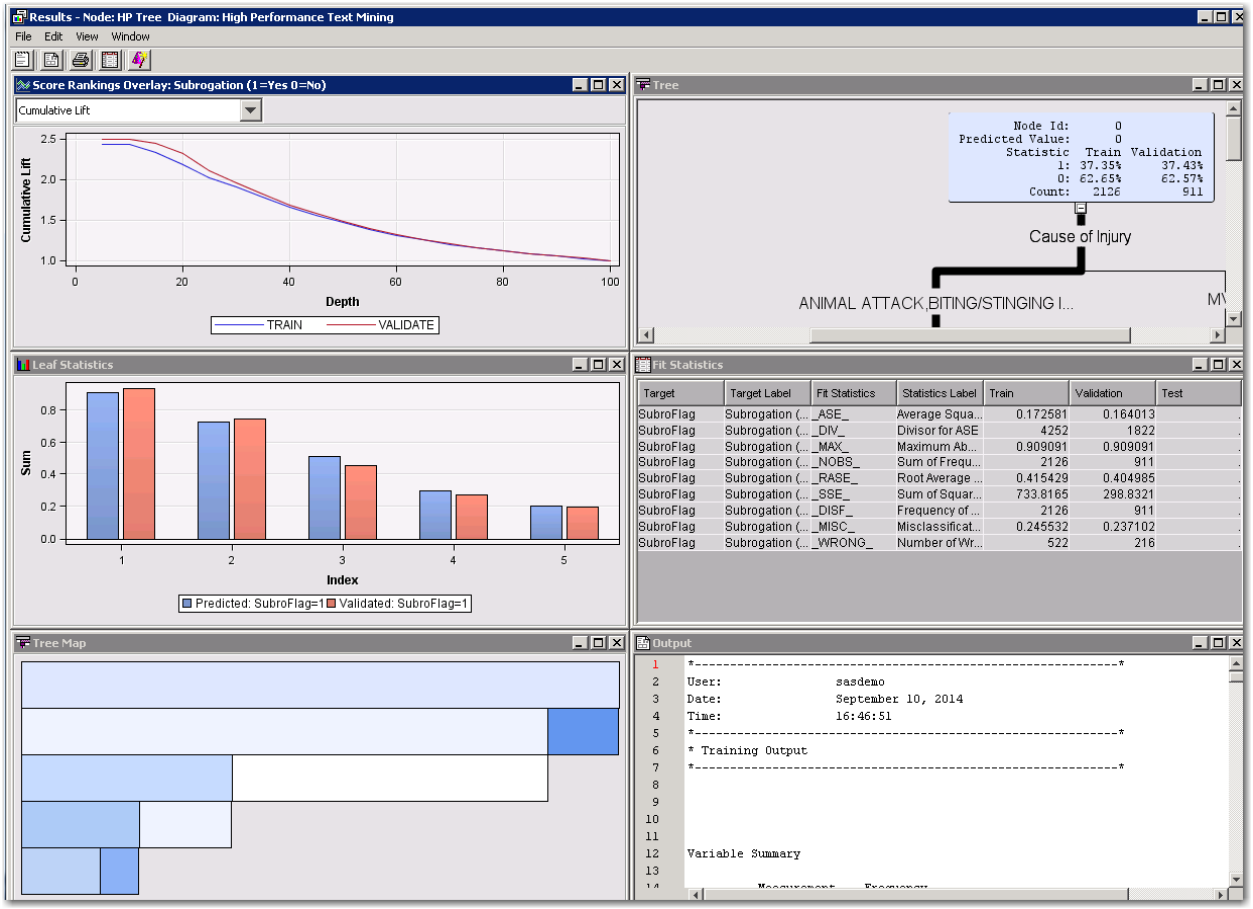


Exercises

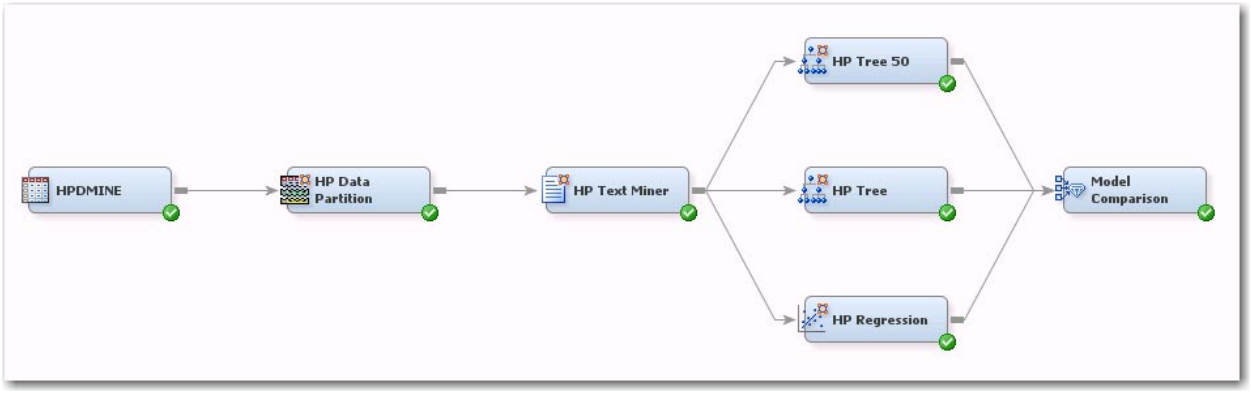
2. Using the HP Data Partition, HP Text Miner, and HP Tree Nodes

a. Duplicate the instructor demonstrations for this section on high-performance nodes.

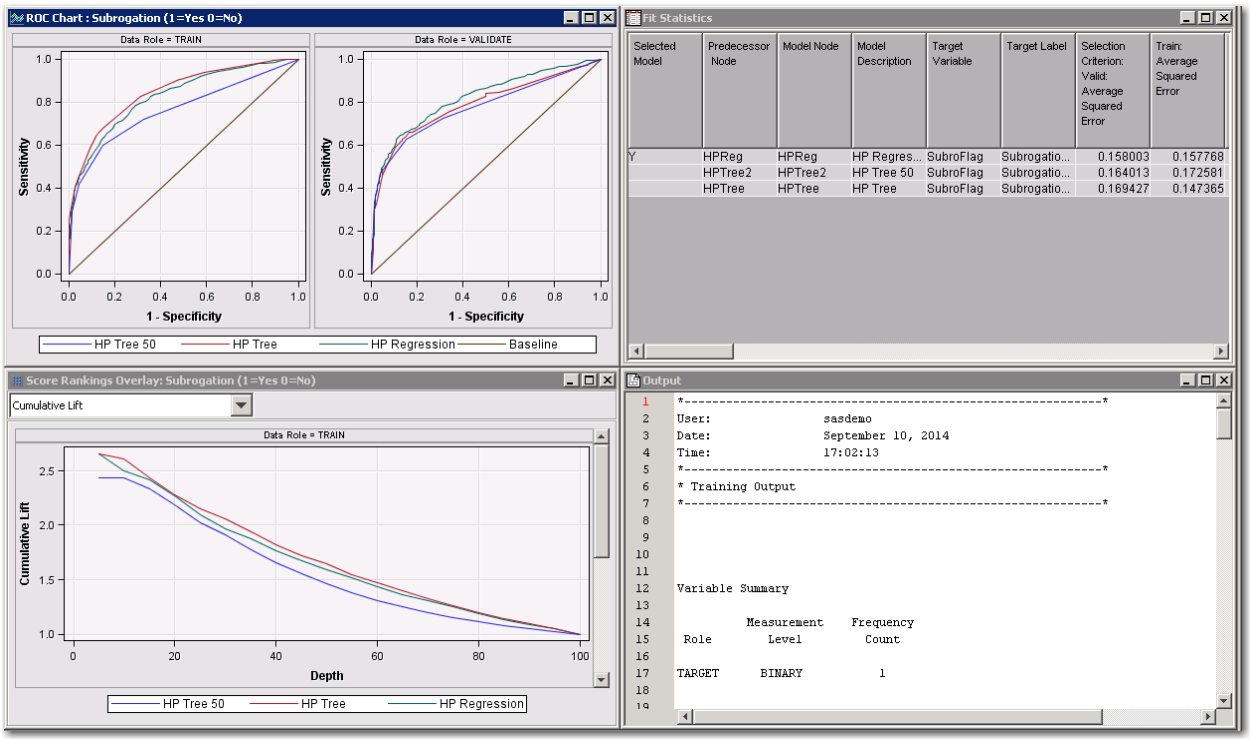
b. Connect another HP Tree node and rename it to **HP Tree 50**. Set the Leaf Size property to **50**. Run the node and compare it to the first tree.



c. Add an **HP Regression** model to the diagram. Set the selection method to **Forward**. Connect a **Model Comparison** node to the three models using **Average Squared Error** on the validation data set as the selection statistic. What is your champion model?



Note the valid average squared error for the three models.



4.4 Chapter Summary

Enterprise Miner has many powerful predictive modeling nodes, as well as nodes that support modeling in a variety of ways. The Text Miner nodes are integrated into this environment. The default settings for the various Text Mining nodes (and other Enterprise Miner nodes) are designed to be good across many situations, but are not always going to lead to the best results for your particular data. Take an experimental point of view and use Enterprise Miner and all its nodes as a convenient "workbench" to test out a variety of different property settings to see what gives you the best models in your environment.

In predictive modeling, there is usually a trade-off between better model performance (as assessed by various metrics, such as the ROC index and the average squared error.) and more interpretable models. A reasonable strategy to manage this trade-off is to create ***both*** types of models: models with the highest possible assessment performance, regardless of interpretability; and models where there is high interpretability. In this way, you can involve management, clients and other interested individuals in the decision of which model should be used for implementation. Both the Text Cluster node and the Text Topic node can be used in ways that can enhance predictive power at the cost of interpretability or vice versa.

The Text Rule Builder node provides a stand-alone predictive modeling solution for data having a text variable and a categorical target variable. This node creates Boolean rules from small subsets of terms to predict a categorical target variable. The node must be preceded by Text Parsing and Text Filter nodes.

High-performance text mining and predictive modeling procedures are designed to take advantage of specially configured computing environments and technology. Distributed configurations can enhance the speed of analysis by running multi-threaded parallel processing and I/O operations. The path length of analysis is shortened even further when running analytical process either in the database or alongside the database. Reducing the number of passes through the data results in less total read time when data is kept in memory providing fast analysis when needed.

SAS Enterprise Miner includes high-performance nodes on the HPDM tab. The High-Performance Text Miner node offers simplified selections and combines tasks performed by several individual Text Mining nodes. Text parsing, filtering, and topic creation all are accomplished in the one node. The results can be combined with additional data for supervised predictive modeling applications.

References

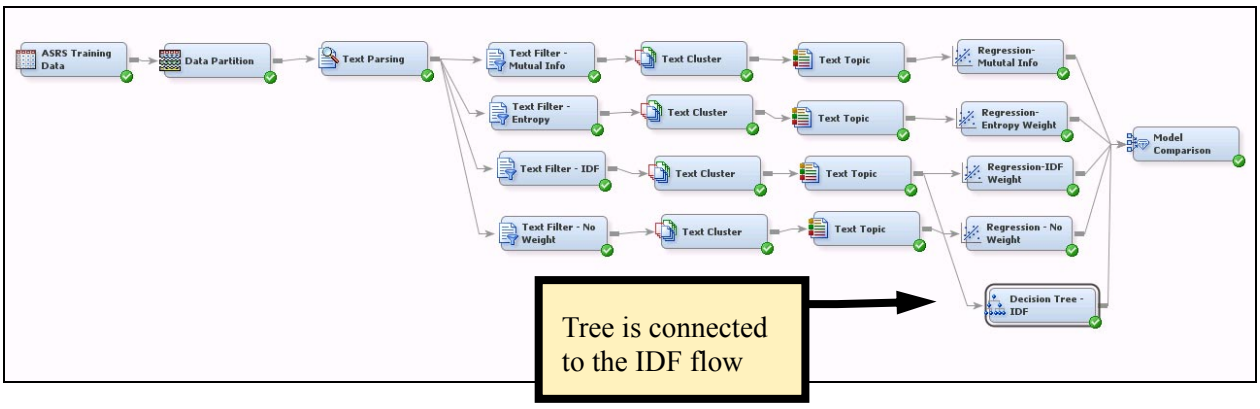
E. Allan, M. Horvath, C. Kopek, B. Lamb, T. Whaples and M. Berry. Anomaly detection using nonnegative matrix factorization. In M. Berry and M. Castellanos, Editors, *Survey of Text Mining II: Clustering, Classification, and Retrieval,* pages 203-218. Springer-Verlag London Limited, 2008.

4.5 Solutions

Solutions to Exercises

1. Adding a Decision Tree Node to the Flow to Compare to the Previous Four Regression Models

The tree should be added to the previous flow in this as shown here**:**



The Model Comparison node shows the following results for the Decision Tree node:

Model Description	Target Variable	Target Label	Selection Criterion: Test: Roc Index
Regression-IDF Weight	Target05		0.963
Regression-Entropy Weight	Target05		0.962
Regression - No Weight	Target05		0.952
Decision Tree - IDF	Target05		0.944
Regression-Mututal Info	Target05		0.929

Tree ROC Index for Test Data

Solutions to Student Activities (Polls/Quizzes)

4.01 Short Answer Poll – Correct Answer

What are the Test Average Squared Error values for the four models generated with different term weights in the demonstration?

From the Model Comparison node (dragging the column from the far right):

Model Description	Test: Average Squared Error
Regression-Entropy Weight	0.049857
Regression-IDF Weight	0.049263
Regression - No Weight	0.052916
Regression-Mututal Info	0.061293

30