Data Mining Using RapidMiner

By William Murakami-Brundage

Mar. 15, 2012

A Bit about Data Science

Data mining and manipulation tends to be classified within statistics and mathematics, it actually draws on the fields of data visualization, computer science, psychology, and information science/information systems. The entire data science field intertwines with data- and knowledge-intensive domains such as medicine, public health, epidemiology, genetics/genomics, and health care. Within ten years, it will be impossible to functionally separate data science from the base sciences that it supports. Indeed, the overlap of data and science is sometimes called 'informatics'.

Data science revolves around certain techniques and approaches to data; these methods are not purely science, and often involve more synthesis, deduction, and induction. Synthesis is the act of merging and combining processes and results, and induction is the scientific method of drawing conclusions in a top-down (induction) and/or bottom-up (deduction) approach. Working with data requires a solid logical model, an understanding of mathematics, and technical ability. The best data scientists have a background with both information technology and social, biological, or medical science. As the data manipulation/data mining field is so fresh, the fundamental skills are often developed on the job, in practice.

Chapter 1: Using Clustering in RapidMiner: A Hands-On Approach

By William Murakami-Brundage

Mar. 15, 2012

Clustering is a data mining method that analyzes a given data set and organizes it based on similar attributes. Clustering can be performed with pretty much any type of organized or semi-organized data set, including text, documents, number sets, census or demographic data, etc. The core concept is the cluster, which is a grouping of similar objects. Clusters can be any size – theoretically, a cluster can have zero objects within it, or the entire data set may be so similar that every object falls into the same cluster. This would be rare: most often, objects will naturally cluster due to mathematic and statistical similarities. In the case of text analytics, objects will often cluster due to keywords, phrasing, and subject/context.

In order to really perform data mining, a functional, high-quality data set is needed. For this tutorial, I am using a data set from the World Bank's Data Repository (http://data.worldbank.org), as they allow the public to download and use their data. The concept of sharing data sets for public use is a lynchpin of the open-data initiative, an international philosophy related to the open-source movement. In order to locate your own specific data set for use, please see Appendix I, an annotated bibliography of data sets, sources, and visualization tools. The specific data set used here is the Education data set, but any large, clean data set will work for data mining. Keep in mind that there is a minimum functional limitation to the size of data set you can use: if a data set is too small, it can limit results. Also, if a data set is too dirty or ill-maintained, the results must be considered with a level of suspicion or skepticism. As noted earlier, clean data means high-quality results – and dirty data yields less than stellar output.

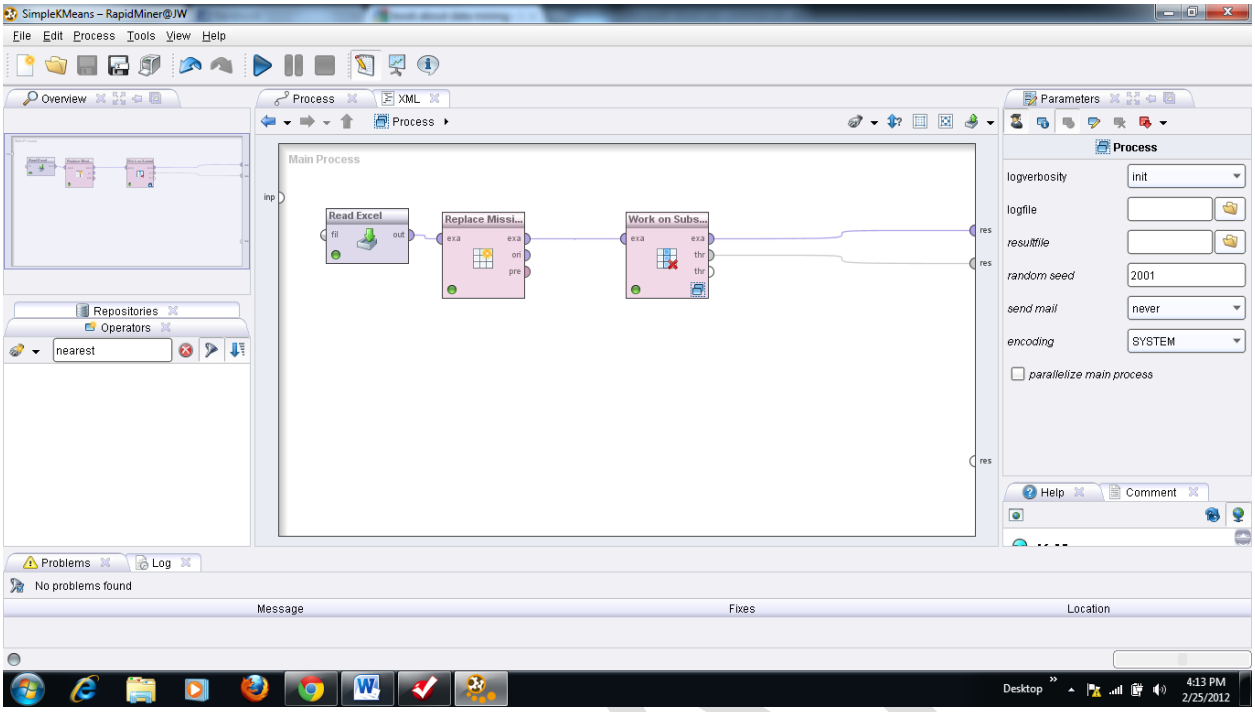


Image 1. Simple K-Means Clustering Method: Overview. This is the overview of the K-Means clustering method. This method will work for most data sets – this particular data has four different labels and 52 columns of numerical data. The clustering algorithm will take this data and cross-compare it in order to group the data set into specific clusters of related items.

Clustering is a great first step to use when looking at a large data set. In order to perform clustering, some setup is required. First, the data set must be prepared and cleaned (Replace Missing Values). Second, the numerical data must be separated into a subset (Work on Subset). Third, the clustering algorithm must be defined and applied (Clustering). Lastly, the output must be examined in order to check for quality and usefulness.

There are many different types of clustering algorithms. Some of the most advanced methods in 2012 revolve about support-vector models (SVM), the CLOPES and COBWEB algorithms, or clustering by expectancy. Unfortunately, these clustering methods require an intense amount of computing power. The K-Means algorithm is the simplest clustering method and also probably the most efficient given limited technology. It may not be cutting edge, but the results are still valid and useful for any data miner looking for the broadest of insights.

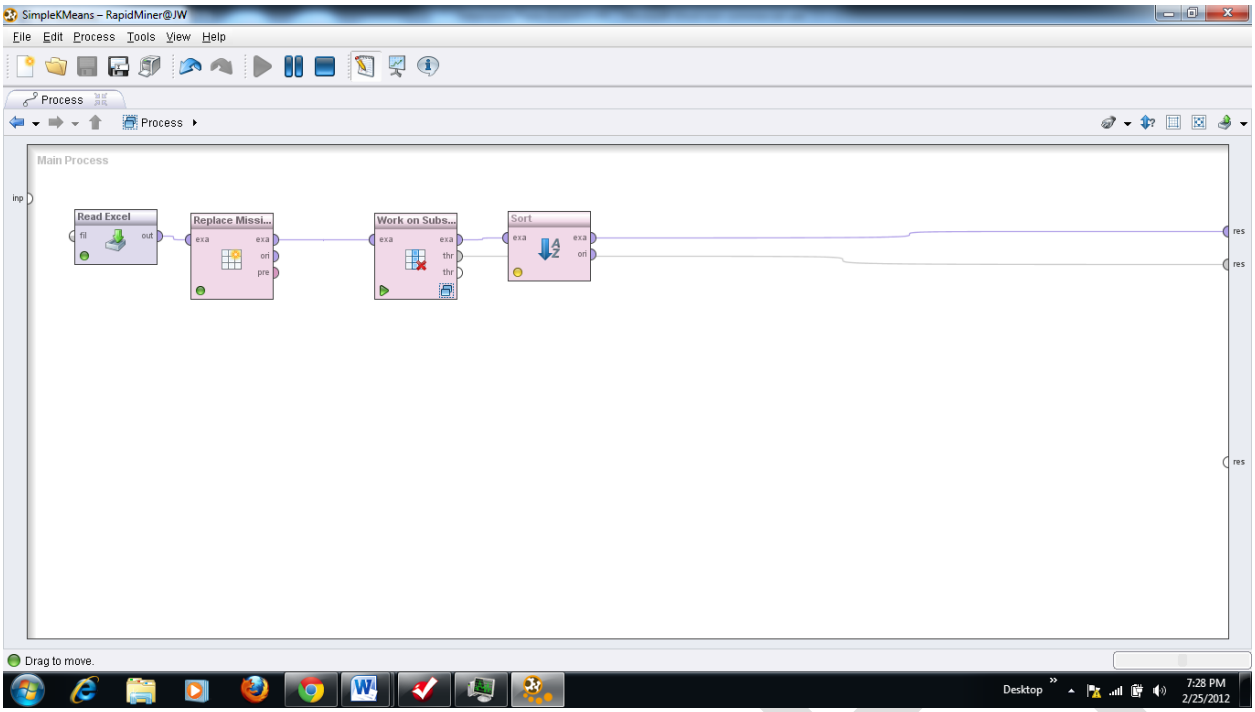


Image 2. K-Means Clustering Process with Sort. This is an alternate process overview, with one addition: the Sort. Sorting allows output to be arranged in descending or ascending order. Any variable can be sorted. While the most common sorting method is alphabetically (A-Z), or numerically (0-9), there can be occasions where data will be sorted in reverse order (Z-A) or (9-0). Keep in mind that the sort command works from the first character in: this can cause confusion when numbers start with a zero (0), or the length of a number is important. For example, if 100, 1000 and 1001 have significantly different meanings, it may be best to find an alternate way to sort them – because they will be placed very near each other once the sort is performed.

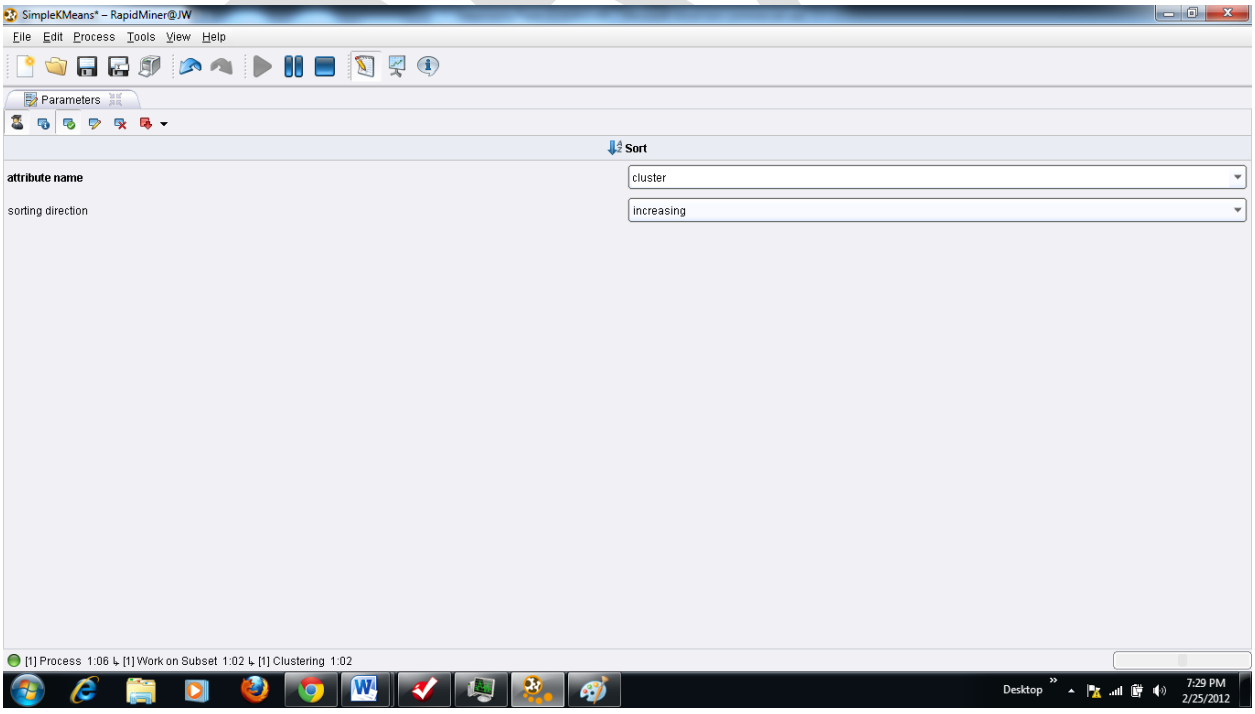


Image 3. Sort Parameters. In case you want to organize your data by cluster, these are the Sort parameters that you would use. Note that there is, as promised, the option for 'sort direction'. The attribute name in this example is 'cluster', but theoretically you could also sort by country code (for census/demographics), provider number (medicine), diagnosis code (epidemiology), college (education), or any other given attribute. Importantly, the attribute must already exist in the data set. As far as I know, most data mining software have limited ability to assign new attributes and labels. This is one reason that it helps to have a high-quality data set in the beginning of your work – it is easier to correctly define decisions such as 'what do I sort on?' and 'how do I label this cluster?'

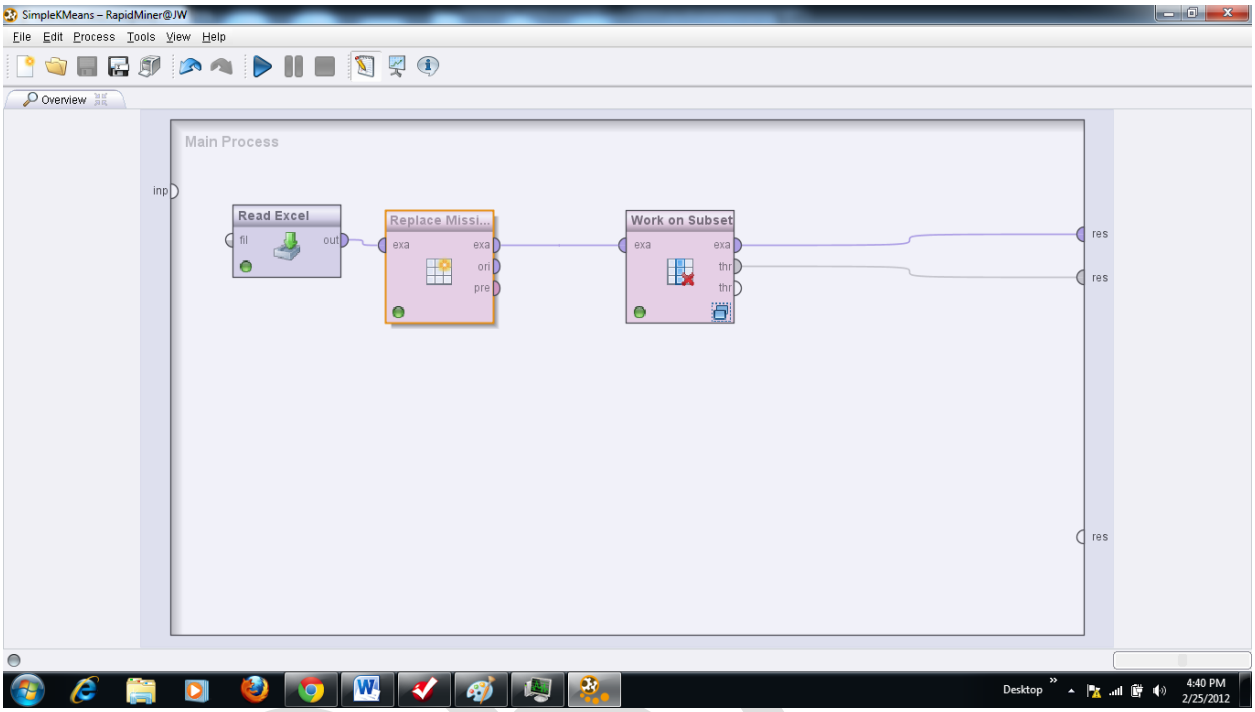


Image 4. K-Means Clustering Process Overview, without Sort (Pareto). This is an expanded view of the Simple K-Means process, in order to show RapidMiner's GUI in all of its glory. You can see the connections running from Read Excel, to Replace Missing Values, to Work on Subset, and then two connections to lead to the output. What you cannot see is the sub-process within Work on Subset. Also, if you incorrectly connect the operators, the process typically won't work. This is actually a good thing, because you want to make sure that your data mining is actually valid. Of special note are the two output connections – one tells RapidMiner to display the Example Set output, and the other is an instruction to output the Clustering from the Work on Subset sub-process. Please connect them both, because you want the full result set.

The small blue square in the bottom right corner of the Work on Subset operator indicates that there is a sub-process involved. The green light in the bottom left corner indicates that the process should work correctly. If the process is untested, the light will be yellow; if the process doesn't work, it will be a red light. This color-coding makes it simple to spot large-scale errors fairly instantly; finer-scale errors still need some debugging.

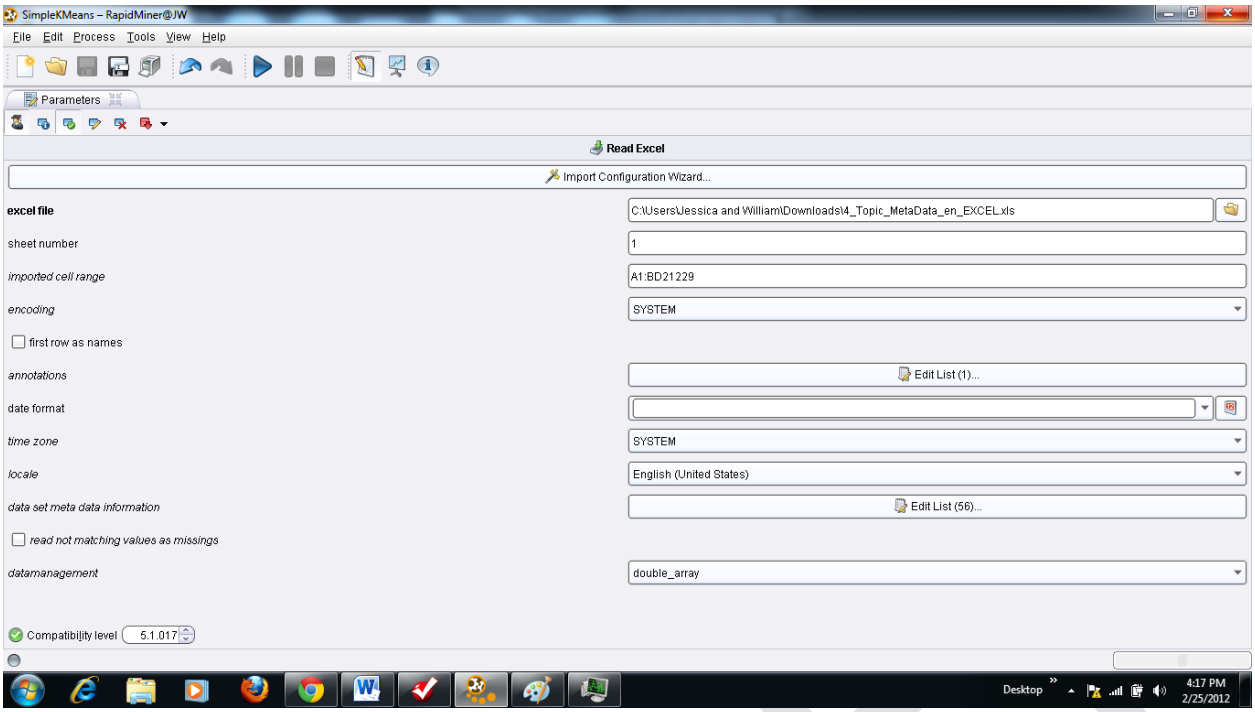


Image 5. Read Excel, Parameters. Data sets come in all shapes and sorts. While there are many databases and API feeds that stream data in a constant flow, the most common data set in 2012 is probably the spreadsheet. I don't expect that spreadsheets will be dominating the data field after the next 10 years – realistically, databases and data warehouses are starting to replace things like spreadsheets and basic data tables. That said, when you are looking for a data set, it is easiest to start with a spreadsheet. The Excel sheet is the de-facto standard for businesses and organizations, but there are other formats out there, such as CSV, text-delimited, and the OpenOffice format. When using CSV and text-delimited data, there are extra steps that necessary before really being able to use the data set.

Shown here are the Read Excel parameters, as they apply to a fairly large, 5.5MB dataset comprised of about 22,000 rows and 60 columns. This is the size of data set beyond simple analysis - you can't just peek and see how to group the variables. Larger data sets are fantastic for data mining, but even a 400Kb data set can yield some insight into the story behind the data. Just keep in mind that there is going to be a lower threshold where the data is suspect – statistically, if your sample is too small, the results cannot be considered 95% accurate. Thankfully, larger data sets don't tend to run into this issue. Indeed, once your data set gets too large, desktop computers may not even be able to handle the computations involved. At this point, you would draw off a random sample, or start using an analytics server such as RapidAnalytics, the bigger brother of RapidMiner. For my system, 10MB is where this upper threshold starts. If you are using an older or less powerful computer, even a 3MB file may be too much. This is why I encourage you to find multiple data sets from various sources, as this way you can experiment within multiple disciplines.

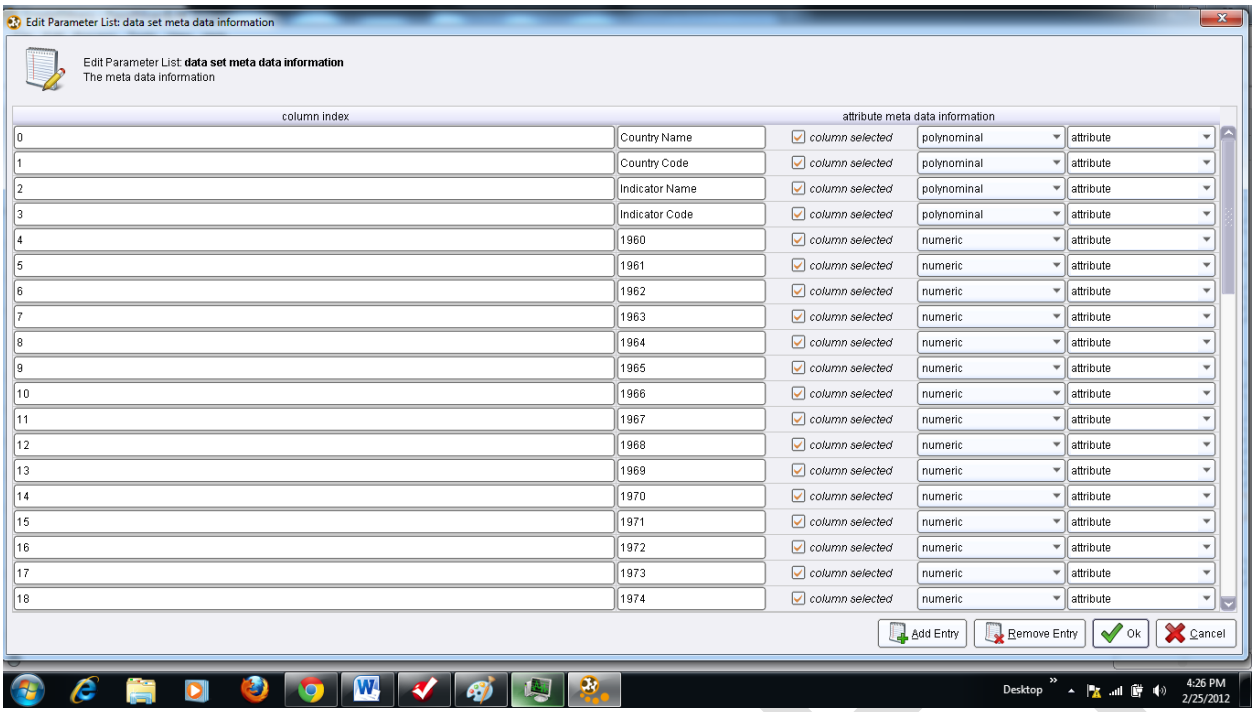


Image 6. Read Excel, Data Set Metadata Information, Column Definitions. This shows the Read Excel metadata for the data set columns. The first four columns are known as the 'labels', and these are the key to understanding what you are looking at. There are two different variables (Country and Indicator), and each variable has two types (Name and Code). The Code is merely a shortened version of the Name. For instance, the Country Name for row 1 is identical to the Country Code, and the Indicator Name for row 1000 is the same as the Indicator Code. Where this becomes more important is when the output is visualized and examined – it is pretty critical to know which country and indicator you are looking at. As far as the Read Excel parameter goes, you just want to make sure that the defining attributes are labeled as 'polynomial' and 'attribute' or 'label'. RapidMiner will sometimes default to 'binomial' for attribute definition, which can cause errors. 'Binomial' means 'two', and 'polynomial' means many. Since this data set has multiple (more than two) entries for Country Code/Name and Indicator Code/Name, you want to select 'polynomial'.

Not shown here are the other 40 or so years that comprise the data set. For the numerical data, selecting 'attribute' and 'numeric' will suffice. Clustering will work with numerical data, so you won't need to define anything as

nominal or ordinal. These attribute definitions have definite uses, but not in this data mining process.

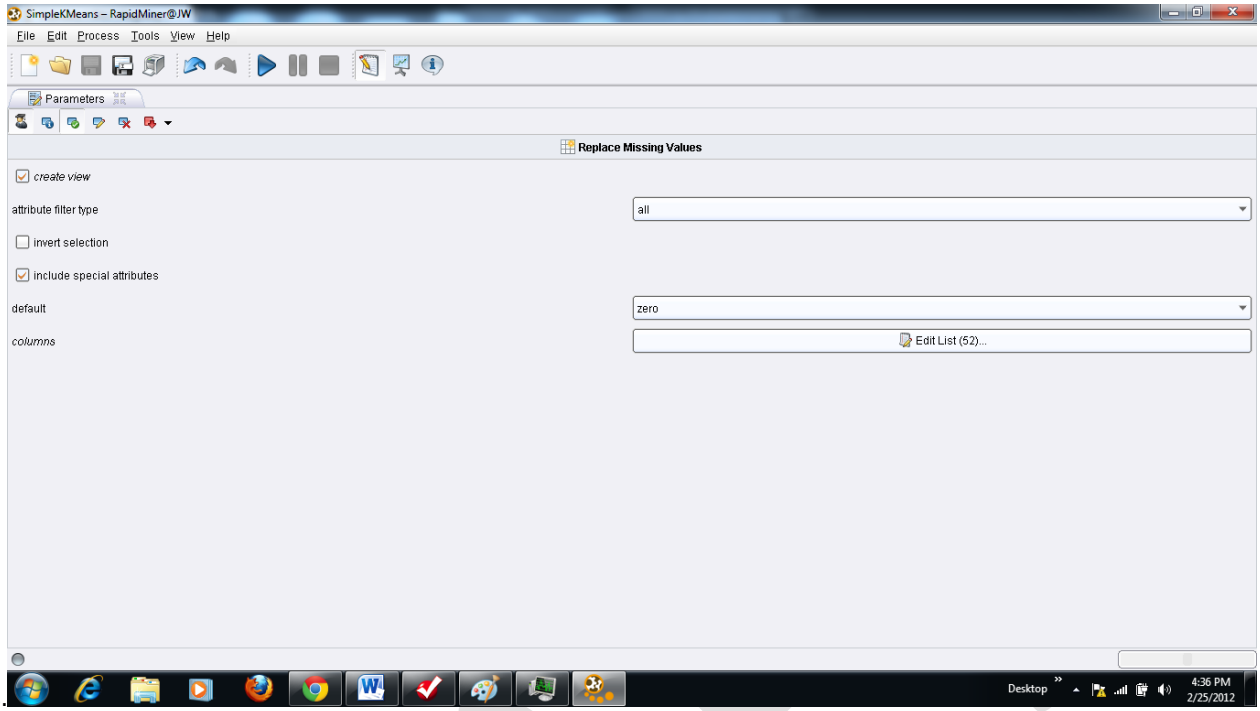


Image 7. Replace Missing Values, Parameters. Here is a tricky bit of work – how to decide what to replace the missing data elements with. You see, often data sets won't work unless they meet certain criteria. For K-Means clustering (and probably for most clustering algorithms) this means that the data cannot have NULL data. NULL data is where the element is blank. There are many raging debates about how and what you introduce into the data set to replace error-causing elements.

When it comes to this data set, I have decided to replace all the blank variables with a zero (0). Other options include replacing missing data elements with the average, the mean, or the mode, or to just leave it blank (not recommended unless you know the process will work while incomplete). There is often a specific, right answer to these questions, but not always. As the data analyst, you need to weigh the impact on your results when substituting in non-original data.

The justification for substituting zero is because the data is not present, and the data set deals with a large number of indicators. While it could be (and probably is) more statistically accurate to substitute an average or mean, it is unlikely that there is a universal answer (i.e. some indicators may use an average, and some may use the mean or mode). Thus, the best answer, in my observation, is to use zero. This at least guarantees that every blank data set element has the same universal answer. In this scenario, zero means expediency and completeness of the data set, while sacrificing possible statistical accuracy.

One option (not explored here) is to re-run the data mining process and examine the output from different missing value replacements. Thankfully, the process should continue to provide output. Don't be surprised if the output doesn't change much with a different value replacement. If all your numbers are close to each other, then the average won't impact the results drastically. On the flip side, if you decide to substitute zero for a data set comprised of large numbers, you will indeed change the results to some degree. This data set I am working with has a mix of large (10 digit) numbers, percentages, and years of education received (with a range from 8 to 24 years).

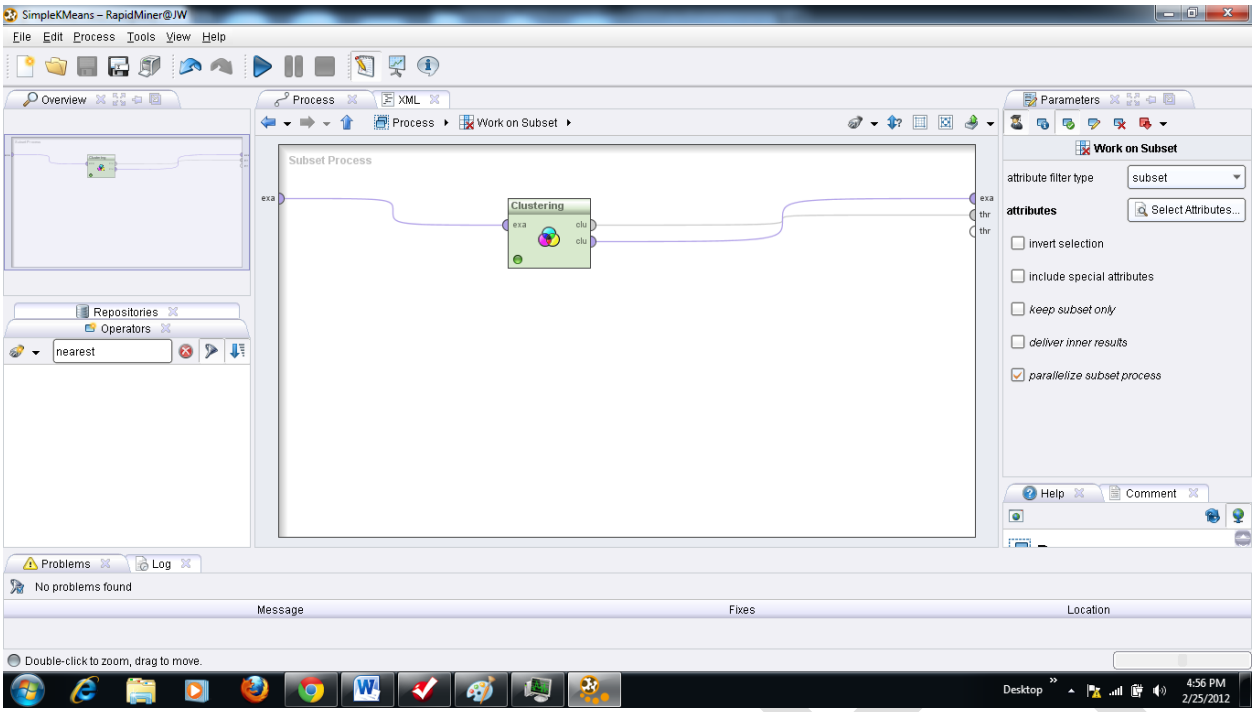


Image 8. Clustering, Process Overview. Drilled down from Work on Subset. This operator is where the clustering work is actually done. The rest of the data mining process is preparation for this operator, which breaks down the data set into the clusters.

In this operator, make sure that you connect both the output connectors from the Clustering operator. That way you know that you are getting the full result set. Also, pay close attention to the cluster parameters in Image 9 . They can show the details about how you may want to set up your clustering operator.

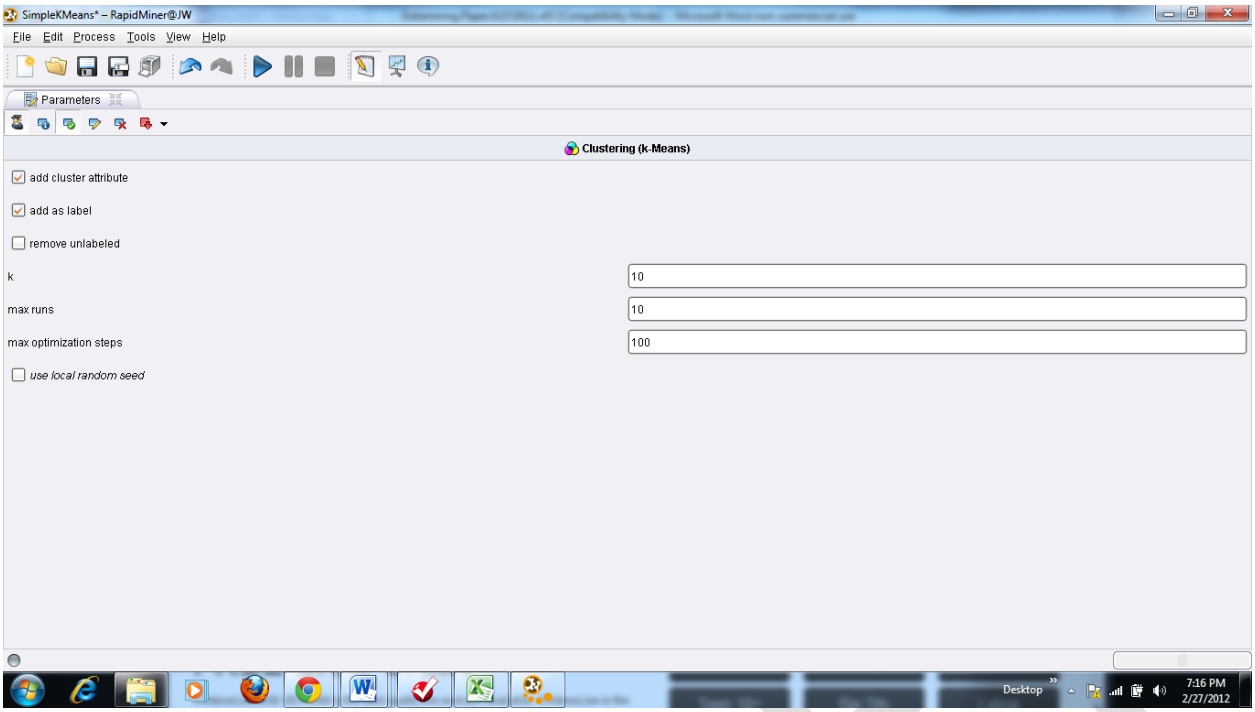

SimpleKMeans* – RapidMiner@JW
File Edit Process Tools View Help
Parameters
Clustering (k-Means)
add cluster attribute
add as label
remove unlabeled
k
10
max runs
10
max optimization steps
100
use local random seed
Desktop
7:16 PM
2/27/2012

Image 9. Clustering, Parameters. So within the Clustering parameters, there are few things to note. The K variable is primary variable, and tells the process how many clusters to create. For our example, K = 10 (10 clusters). In theory, you could have anywhere from 2 clusters to 100 clusters or more. There is no real upwards limit beyond what your computer system will handle. Max runs and max optimization steps are a little bit trickier, and actually show the whole importance of data mining vs. typical statistical analysis. Max runs tells the process how many times to run the results, and max optimization is how many times each run is optimized. This is where the power of data mining and computing meets mathematics and statistics in a rather atypical way – the results are run and rerun, optimized hundreds of times, in order to ensure that the output is accurate.

When you manually calculate a data set such as this, it can be nearly impossible to run a data set 1000 times in order to ensure optimized results. With RapidMiner, or any other major data mining software application, it is almost assumed that you will optimize the results to this degree. As a matter of fact, 10 runs of 100 optimized results is probably a little low, but the results are consistent. Statistically, the chances of error decrease with each optimization that is processed – therefore ensuring better, higher quality output. For the note, there is an upwards ceiling for output quality – don't think that 1,000 runs of 10,000 optimizations is necessary, because it probably isn't. You won't impact the results negatively for doing this, so find a happy level where your computer system doesn't churn away for hours, and you can accept the results.

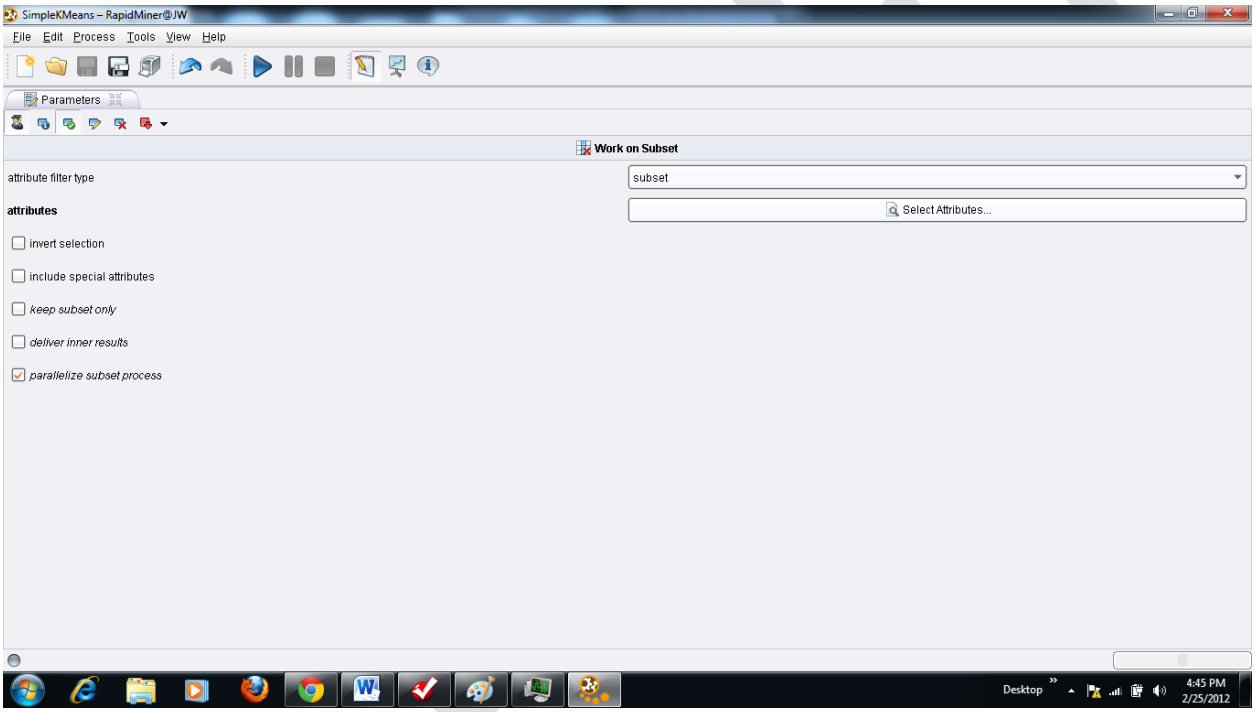


Image 10. Work on Subset, Parameters. Using the Work on Subset operator, you can define the subset that the process will analyze. This isn't just fluff, because often operators will give an error message with a properly defined subset. For the clustering algorithm, a basic tenet is that the data must be purely numerical, and without any missing data elements. We took care of the latter by using the replace missing data operator; now we are going to separate out the numerical data, process it, and then return it into the original data set. The result can be seen down below in the Output Images (Images 10-23). The reason for this is that the K-Means Clustering operator cannot handle polynomial, binomial, ordinal, or nominal data – only numerical. Perhaps other clustering algorithms can work with ordinal or nominal data, but for our purposes, everything must be numerical.

Noteworthy: The subset and clustering doesn't care whether the numerical attributes are percentages, huge numbers, decimals, or whole numbers. The data just needs to be in the form of numbers, but beyond that, the whole process is fairly format-agnostic (it doesn't matter if one column is a percentage and other column is a 15-digit whole number). This is one of the major strengths of data mining applications – the overhead to prepare the data set and get results is far less than the overhead to manually calculate a fraction of the output. Really, mathematics can no longer be separated from technology – the two are completely inter-dependent.

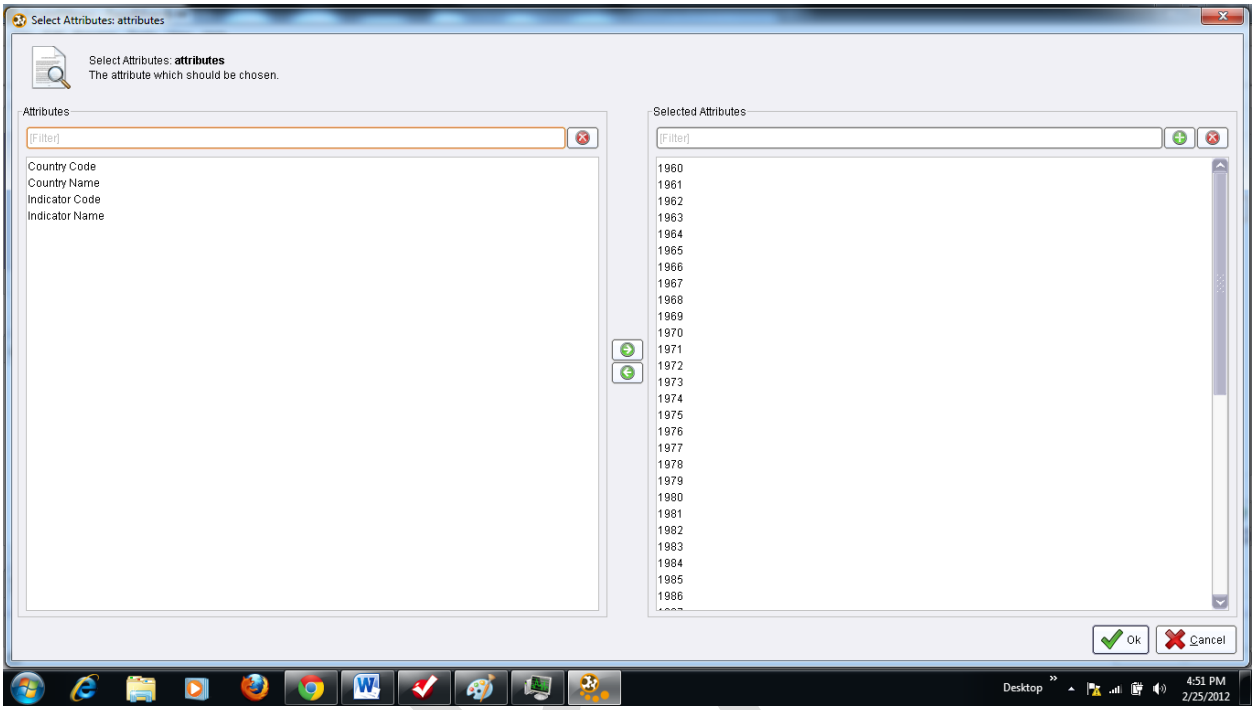


Image 11. Work on Subset, Select Attributes. Here is a demonstration of how the attributes are selected. On the left, we have the non-selected attributes. On the right, we have the selected attributes. In the middle are the arrows that move attributes back and forth between the columns. The goal when selecting this specific subset is to put all the numerical attributes on the right, and leave all the non-numerical attributes on the left. It is as simple as that.

With other data mining processes, you could always partition out your columns and process them differently, but for this data analysis it is fine to just put them into the two categories – numbers and non-numbers.

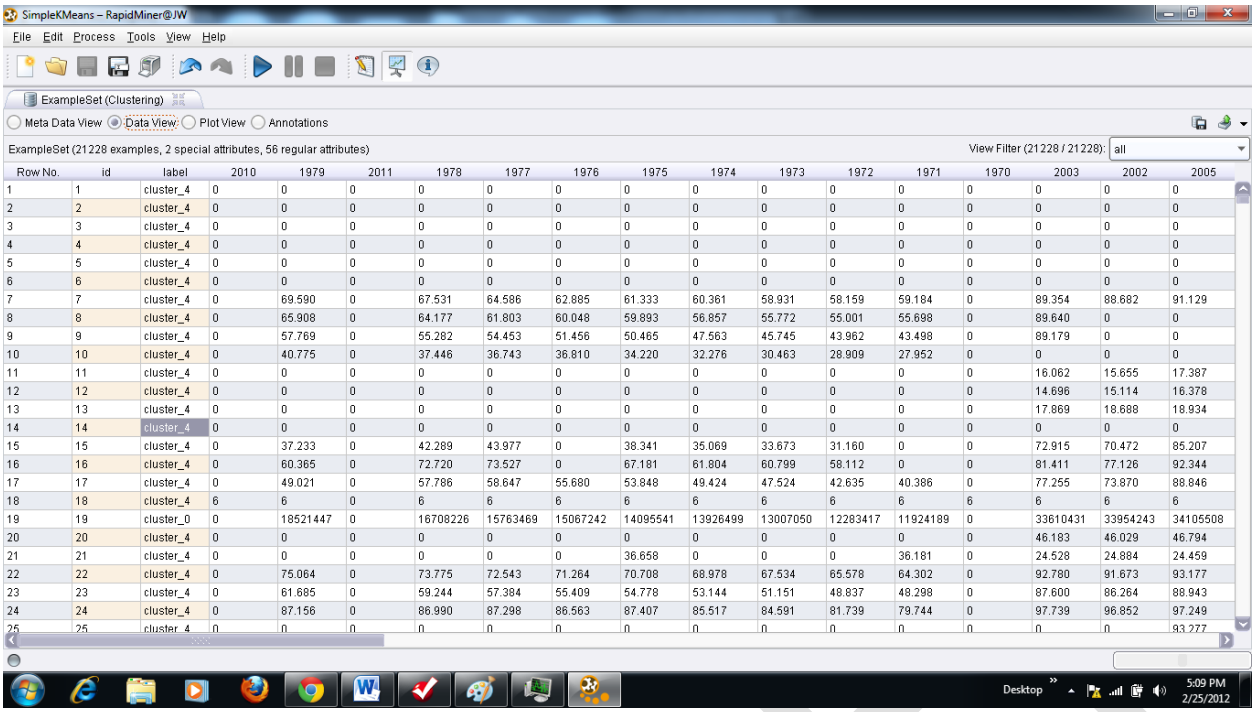


Image 12. Output, ExampleSet (Clustering), Data View. Here is an example of the example set yielded from the output. This is the raw data with the addition of two columns: 'id' and 'label'. The ID column is basically a replicated Row Number column generated by one of the operators. The key result in our output is stored under 'label', where you can see cluster_4 listed for the top results. This shows that the clustering algorithm has properly worked, and that every row has been classified into a cluster.

While this may not seem like a miracle, now that the results are clustered, you can perform data analysis and visualization techniques using RapidMiner's Plot View, which is something that we will do below.

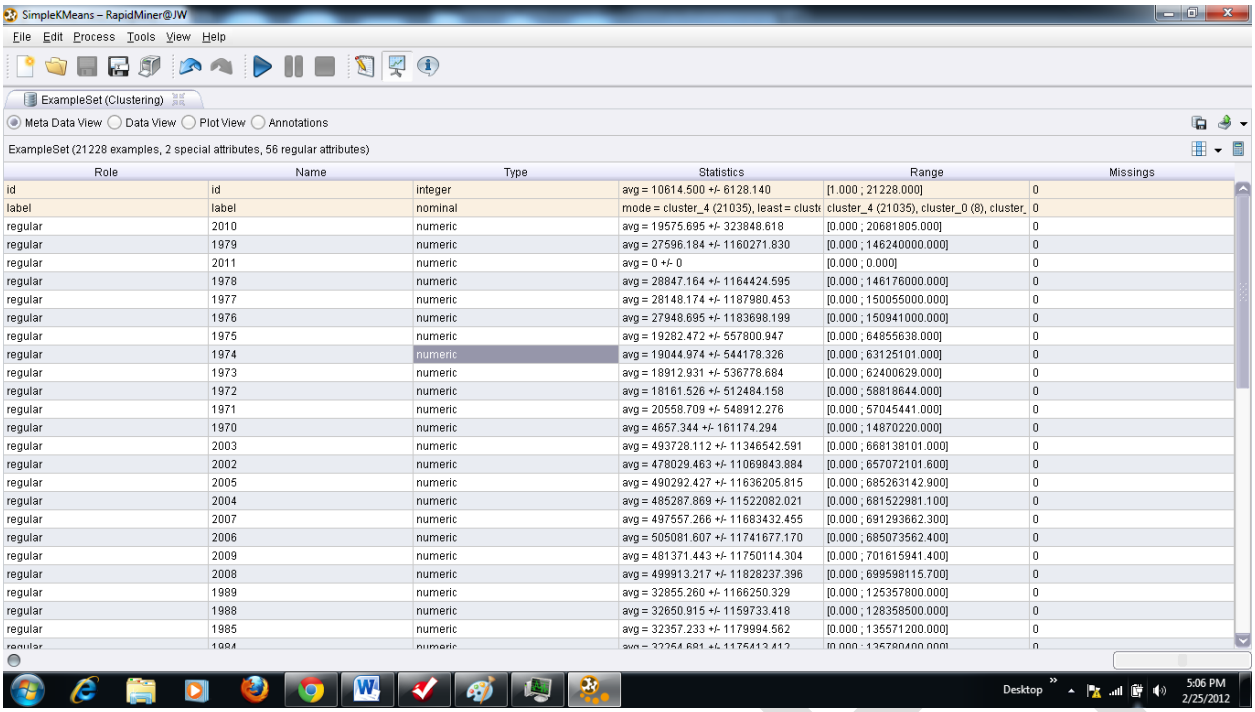


Image 13. Output, ExampleSet (Clustering), Meta Data View. Here is the ExampleSet metadata view, where you can see the two new attributes (id and label), as well as their types. You can also spot the statistics and range – when you take a peek at them, they will show a few things. For instance, you can see that the average and range for the year 2011 is a big, fat zero. This means that there was no data available at the beginning of the processing, and so every column was replaced with a zero when we used the Replace Missing Element operator. The final result is that there is nothing calculated for this year.

You can also see the missing column. 'Missing' shows how much data was missing from the data set when the results were calculated. Since clustering requires all data to have at least a zero, there should be no missing elements. Often, stickier, less organized data sets have missing data, which can affect your result's quality and statistical validity.

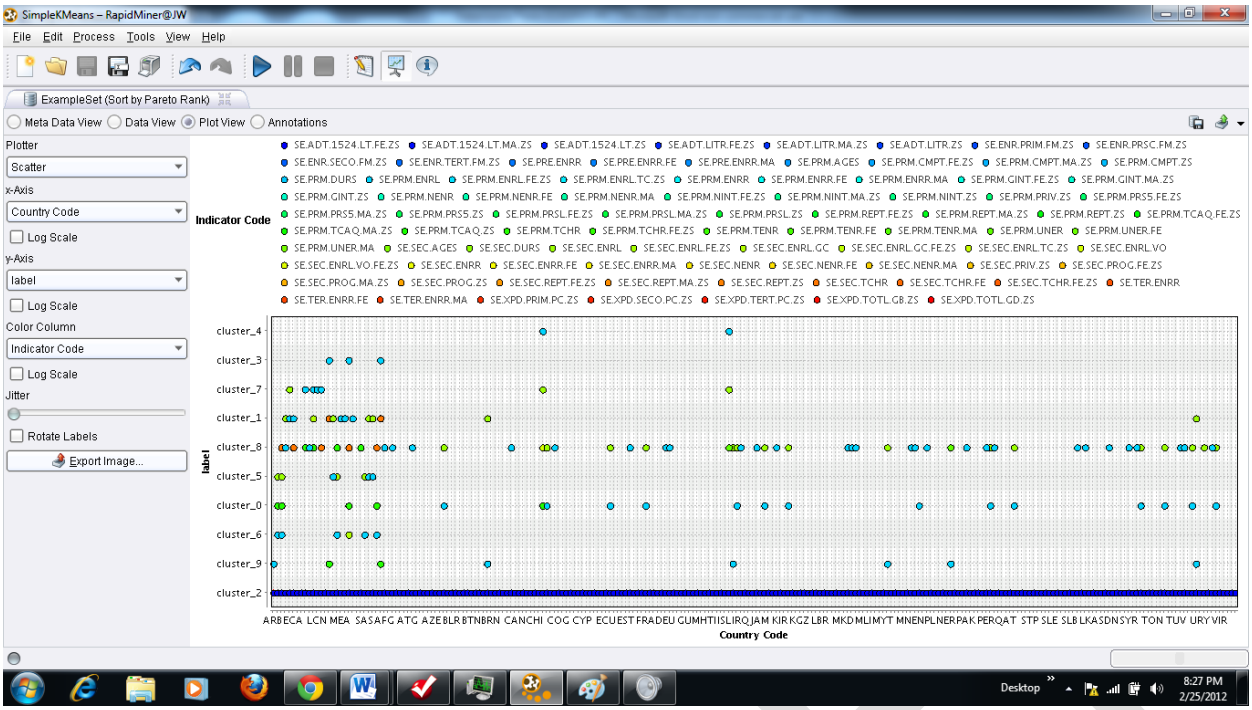


Image 14. Output, ExampleSet (Clustering), Plotting Output View. Here you can see the clusters graphed using the Plot View command. This data is displayed using a variety of techniques in the next few pages, but this is a key way to determine important indicators and countries at a glance.

Looking here, you can see that cluster_2 (at the bottom of the graph) holds the most data elements. This is not a mistake – data mining in this data set shows the elements that are outside of the norm, rather than within the normal range. You can see that cluster_8 holds the next largest array of points, and cluster_4 (at the very top) only holds two points. By hovering over any given data point, a tooltip appears that will show greater detail. By double-clicking on data point, the complete sub -record of that point will be shown. This is immensely useful when you want to drill down into the data mining results and explore the 'who, when, and why' in your findings.

RapidMiner provides this graph in color, by use of the Color Column on the left side of the Plot View display. This allows an even quicker exploration into the data depths, as every indicator is assigned a distinct color. This color coding essentially allows a third variable to be graphed using the Plot View (Scatter) command. In this case, the indicator is graphed in color, but it would be just as easy to graph the data with a different arrangement and display an entire year's clustering output. Data visualization often relies on things such as displaying results in colors, textures, or sizes in order to portray different findings and accent various critical pieces.

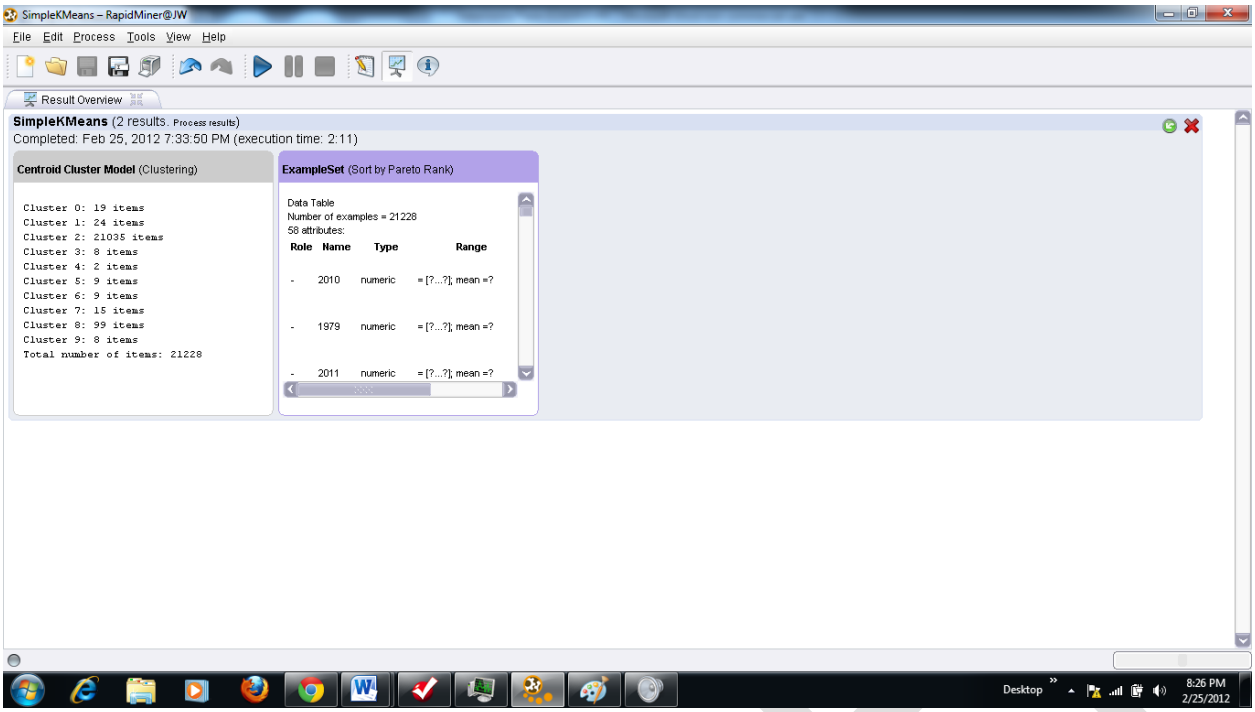


Image 15. Simple K-Means Clustering, Result Overview. Here is the simple output of the centroid clustering model. On the left of the result overview you can see the clustering output (clusters 0-9, with a varying number of elements within each cluster). On the right of the result overview you can see the example set sorted by Pareto Rank. This contains statistics about the output, but doesn't directly impact the resulting graphs or sets.

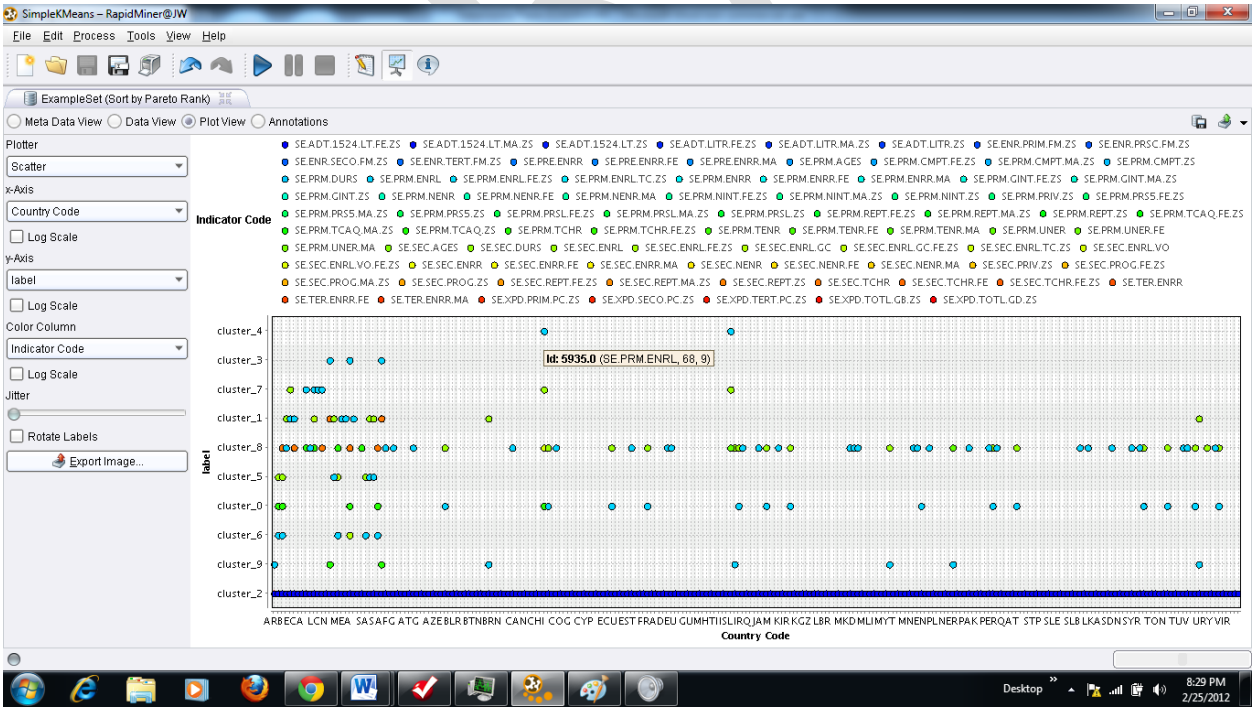


Image 16. Output, Exampleset (Clustering), Highlighted Output. By hovering over any given data point in the graph, you can display the ID number and some of the related data. In this case, it is displaying the ID number, Indicator Code, and numerical data assigned to that data point. It won't directly tell you why the data is in that cluster, which is why you either would investigate that cluster further, or continue to build data mining models in order to winnow out deeper meaning.

Image 17. Output, Detailed View of Data Set Element. This is an expanded view of one of the data points within the Plot View graph (specifically, ID 5935). You can see how this would pretty quickly allow you to summarize and discover new trends and data findings. In this case, the data point corresponds to China's number of primary education pupils that are currently enrolled. This has been grouped in cluster_4, which actually means that is an enormous outlier. Hence, China's school population lies within one of the clusters with only two elements, at the top of the graph.

This is probably one of the limitations of RapidMiner, is that once multiple variables start being graphed and laid out, the output often needs some massaging in order to be presentable. Working with raw data is always harder than just reading results. For instance, cluster_4 may lie at the top of the graph, but may not necessarily be the most extreme cluster. As a matter of fact, clusters will tend to be arranged by their own internal similarities, rather than external factors (typically, elements within a cluster will be more similar to each other than most elements outside of the cluster).

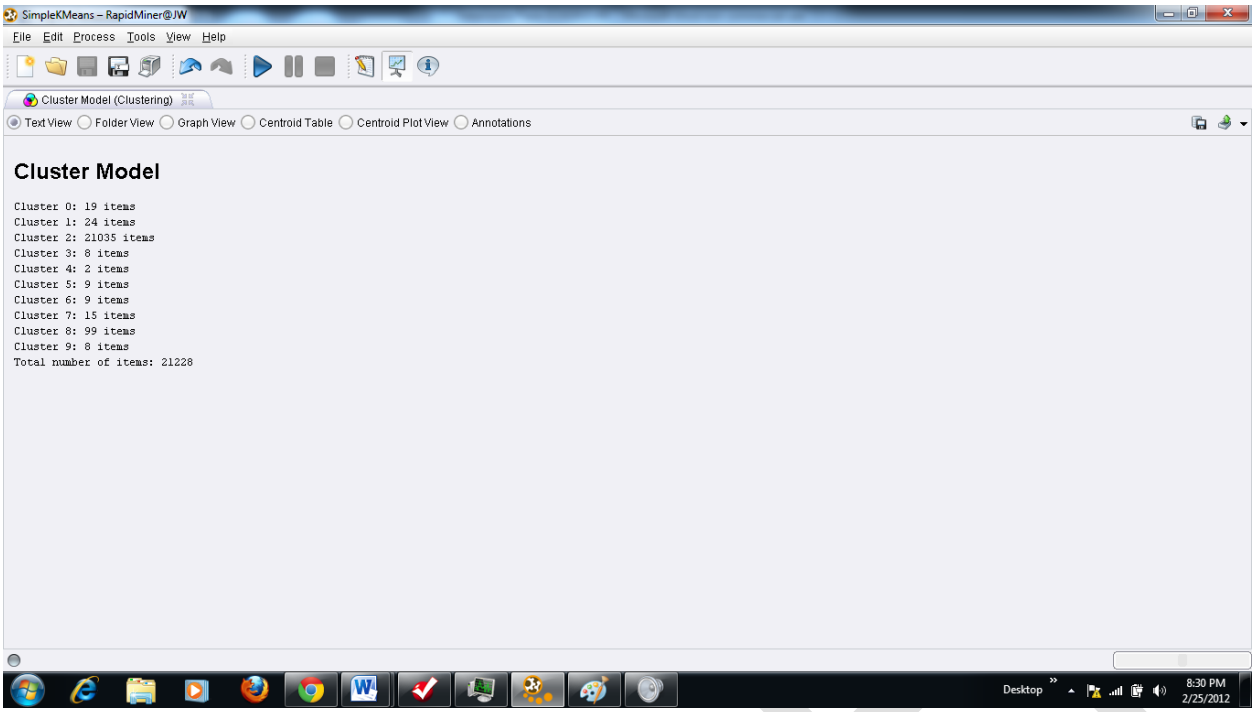


Image 18. Output, Cluster Model, Text View. Shows ten clusters, numbered 0-9. This is the straightforward text output, showing the name of the cluster and the number of elements that each cluster contains. You can see that the majority of the items are sorted into cluster 2, with the next largest being cluster 8. After that, the number of items in any given cluster drops sharply, with cluster 1 having 24 items. The smallest cluster is cluster 4, which we already examined somewhat earlier in the chapter. Cluster 4 has only two items, but there is nothing in the K-Means algorithm that states that a cluster need to have any items. In other words, a cluster can have zero items in it. This is often seen with a K = 2, two item cluster set, where all of the items are sorted into one cluster. It is important to note that an empty cluster doesn't mean output failure, and can actually be a perfectly valid answer.

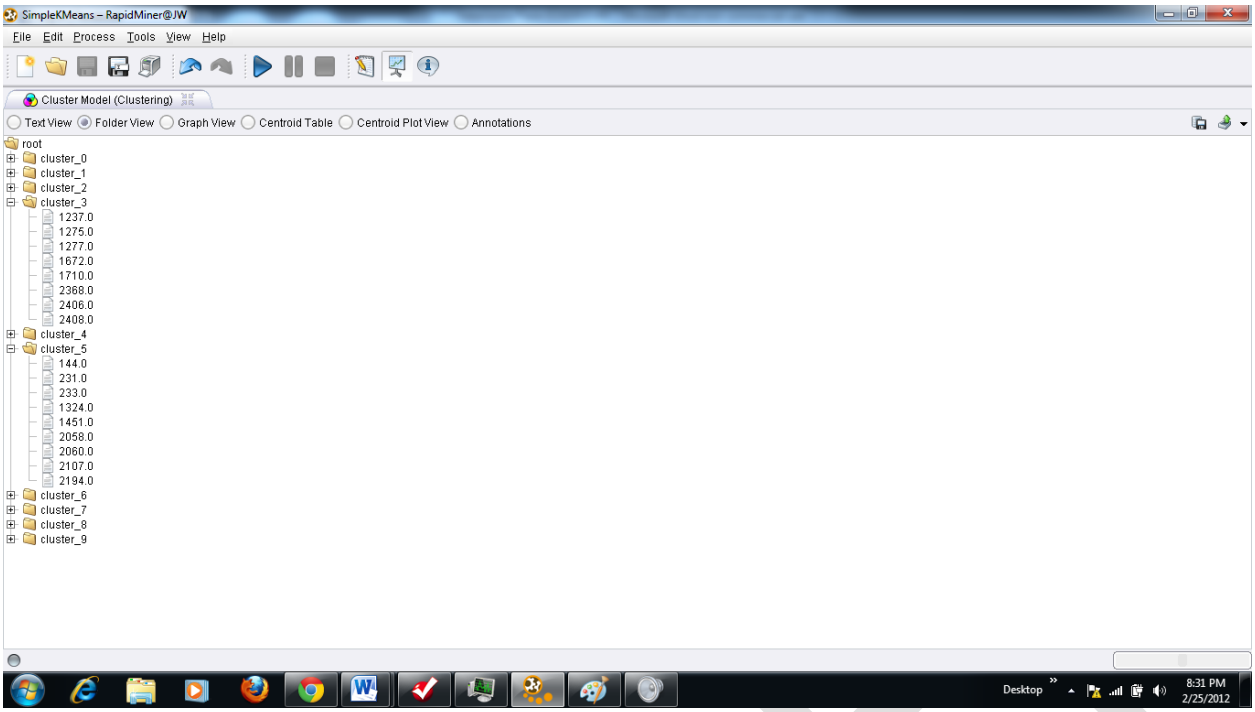


Image 19. Output, Cluster Model, Folder View. This view shows the condensed list of clusters with clusters #3 and #5 expanded to show the data set elements within that cluster. This is the folder view for the Cluster Model, where you can see the actual contents of the different clusters. While this view is not nearly as useful as the Plot View graph, it can still shed some light on how data is arranged. Typically, you would refer to this to see specific data points, and then refer to your data set for further insight. Thankfully, with the Plot View function, RapidMiner enables much quicker data discovery than the 'hunt and seek' method.

One thing that could be performed using the Folder view is to sort out the data set and arrange a sub-set comprised of specific clusters. Typically, this would be done when you are building a data model for machine learning, or a training data set. In this type of situation, you would create a sub-set and run your machine learning algorithms against it, and then cross-compare with your original data set (minus the training set).

It is prime rule that you don't run your training model against your comparison data, because by the very basic rules of statistics you will be skewing your results. Instead, you extract or build a training set from your original data, and at the same time you eliminate the training data from your original data set. This helps ensure that you aren't going to include data which will directly change your machine learning results – because if your training data is a perfect, 100% match for what you are trying to find, then you don't want to find it again! Doing so will make your results look better than they really are.

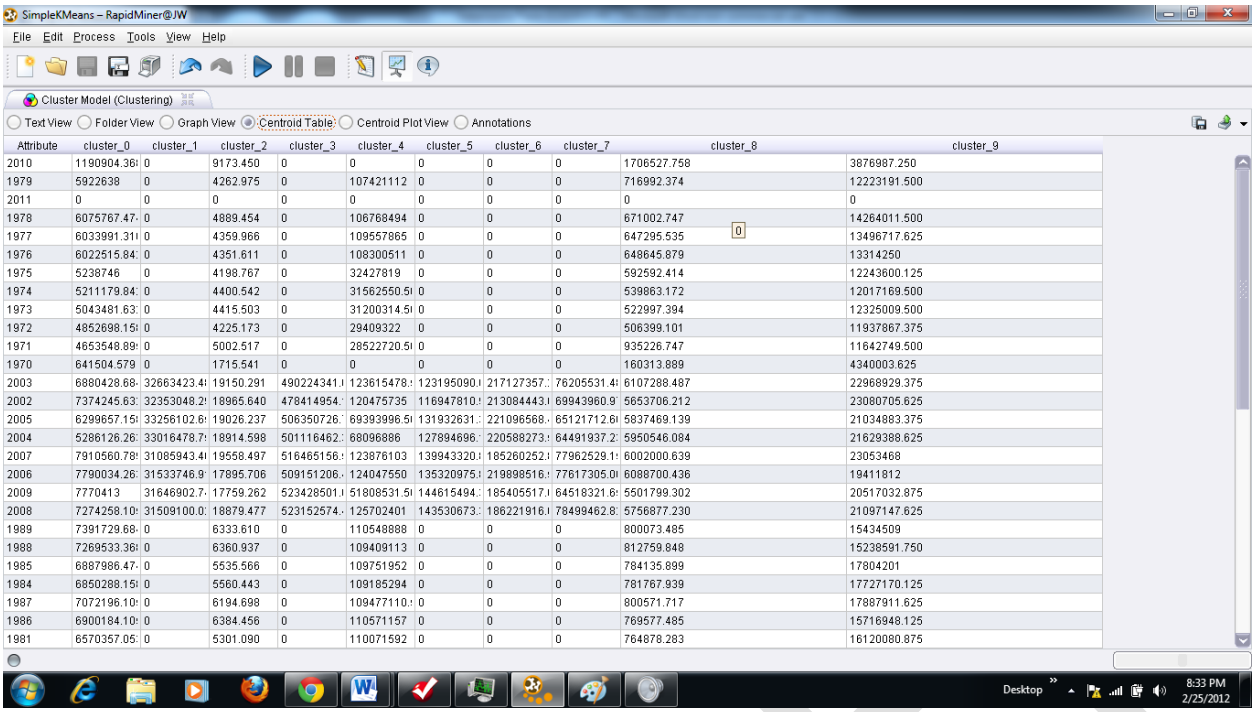


Image 20. Output, Cluster Model, Centroid Table. This shows a data summary for each cluster. It gives the centroid data for each attribute and cluster, which also could help you determine what kind of criteria the clustering algorithm used in order to find your given cluster. For instance, you can see that cluster 2 has a large number of four digit (plus decimal) numbers, which means that the cluster is focused around that size of data point. Cluster 8 deals with much larger, 6 digit data points. Cluster 9 has even larger points, which range into the 10 million. Knowing this can help predict which cluster an unknown element could be assigned to, as well as help form a basic understanding of the data set you are working with.

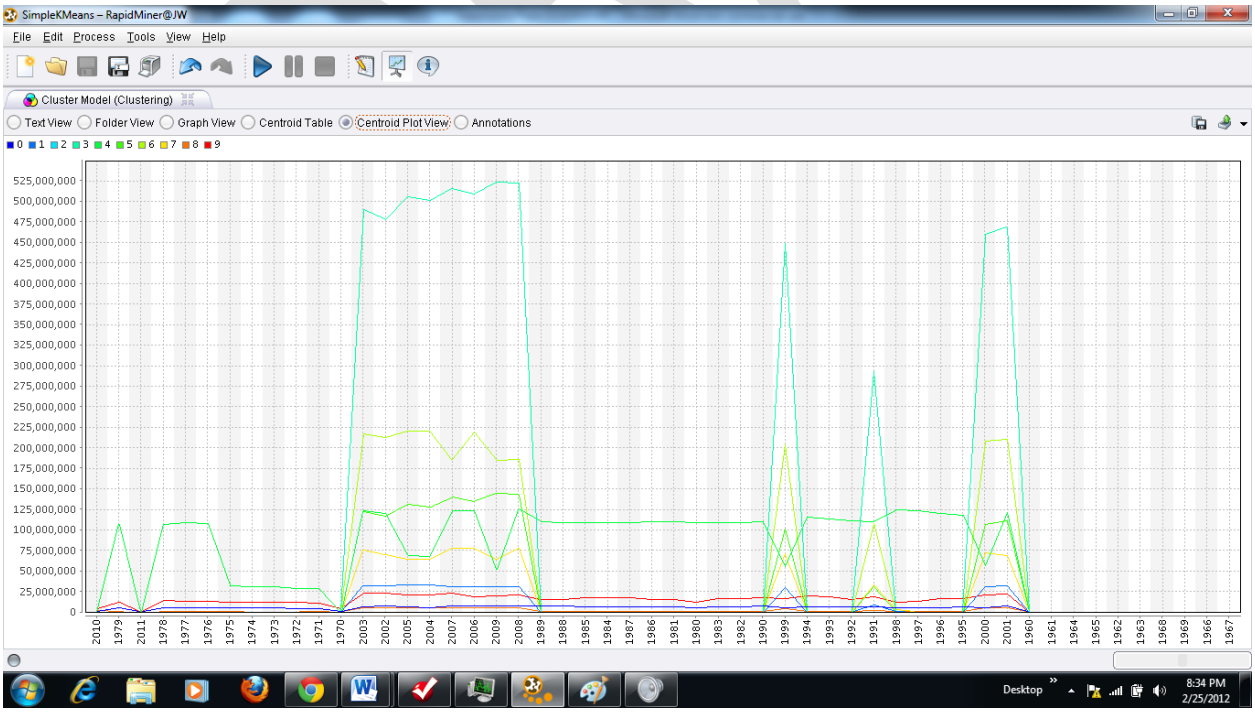


Image 21. Output, Cluster Model, Centroid Plot View. This shows the centroid/numerical display for each year, graphed out as year vs. number. What you cannot see is that the output is color coded by cluster number (0-9) – but if you look you can see the key at the top of the graph. This helps to highlight exactly why a data element will be assigned to a given cluster, as cluster 4 deals with the most immense numbers, and the other clusters deal with smaller and smaller digits. Ultimately, don't forget that data mining is about the numbers, the ordinals, and the data – and numbers will yield up their secrets if you keep looking.

Examining any given data set can take days or weeks. Don't expect an immediate response, but take comfort in the fact that you are doing what was once considered impossible. The fact that we can sift such a huge array of data by just arranging operators and pushing a button is fantastic. Eventually the insight-finding portion of data analysis will also be automated, and then there will be a whole new field of technologist that opens up. For now, keep striving to be at the cutting edge of data science, and I will see you in the next chapter.

Chapter 2: Understanding API Feeds, Data, and RapidMiner: A Hands-On Approach

By William Murakami-Brundage

Feb 28. 2012

At any given point, there are billions of bytes of data flowing within the Internet. Most of this data is indirectly accessed, if at all – information systems have just begun being designed with the idea of continuous capture and temporal querying in mind. On the other hand, there is a vast amount of organized data available in API feeds and other organized databases - an API feed is a direct link to a company's repository, storefront, and inventory.

Etsy.com is kind enough to grant access to API feeds for developers: Amazon, Google, the World Bank, and other institutions also allow users to access organizational API feeds. This use of the Etsy API is performed under their open-access philosophy, as well as my desire to teach others the rudiments of data access within a web application framework. It is my hope that perhaps a small seed will take place in some of my readers, and they will be enabled with a passion for technology and programming, data and systemic design. As a matter of fact, when you look around you can find dozens of API systems available for use; people truly want to encourage budding and established developers. Don't restrict yourself to replicating this work, but instead use it as a reference point for programming and data usage.

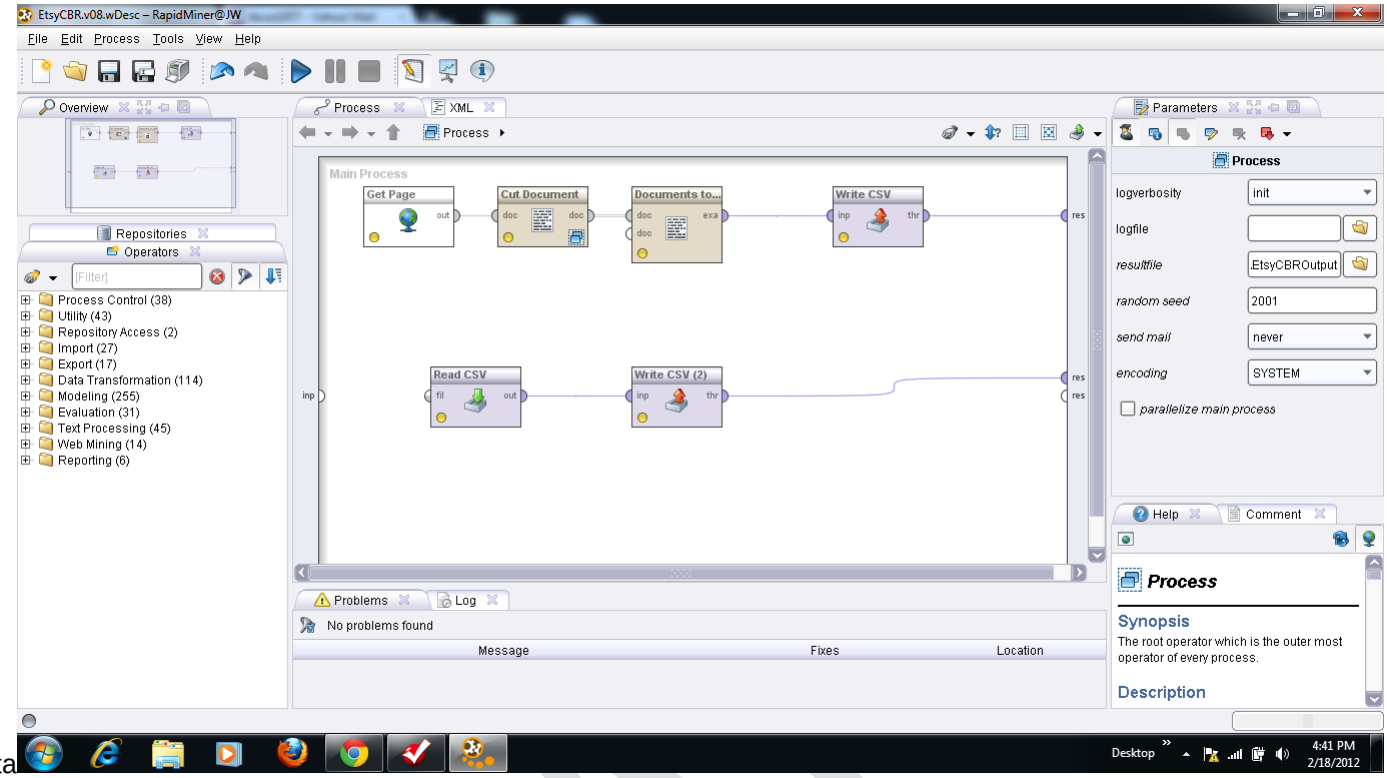
Importantly, Etsy retains all copyright to their data and API at all times.

On Facebook, I once said that all anyone needs to get started with information technology is a large hard drive, an Internet connection, and a willingness to wipe everything out and start over from scratch, again and again, until results start coming in. Also, Seth Godin once wrote a brief bit that really stuck with me about 'poking the box'. In many ways, information systems are the classic black box experiment – the goal is to keep poking the box until something happens. In the rush to become good at something, we often overlook the processes and methods that allow our internal learning to happen.

RapidMiner 5.1 is an open-source system that is available for download from http://rapid-i.com. It is easy to install, the license is free, and there are both a substantial market and community support. Most of my data mining takes place within RapidMiner, as it is on par with any commercial product, but the only price tag is my own effort. In order to process text, you will need the Text Extension; in order to perform web operations, you will need to install the Web Extension. Both of these are free and can be installed from within the main program (from the main menu, select "Help", and the "Update", and then install the required extensions). Currently, all extensions in RapidMiner are free as well. Other extensions include R (the prime statistical package), Weka (a data mining application), Reporting (for reporting services), and numerous others.

It is my hope that by studying this example, perhaps signing up for an API feed or downloading a data set from one of numerous open-data sources, readers can get down into the data and begin to understand how things flow within the Internet. After all, it is 2012, and William Gibson's world is becoming more real every day…

Section 1. Understanding the



Data

Image 1. This is the global overview from RapidMiner 5.1, as it is set up to extract data from an API feed and parse it into an spreadsheet. There are endless reasons that a collection of data points would be extracted and stored in this way: typically the data would be stored in a database such as Microsoft Access/SQL Server, MySQL, or Oracle. For simplicity, this example is made to just extract to a CSV, the most generic format of spreadsheet available.

The top process consists of a few simple steps: it fetches the API feed, cuts the API feed into pieces using a sub-process, converts the separated API portions into data, and stores the results within a CSV file. A variant of this (not pictured here) stores the data within an Excel spreadsheet or database, as noted above.

You will notice that there is a second process that consists of a Read CSV and a Write CSV down at the bottom of the main process: this is no mistake. This secondary process cleans the data an extra step while retaining the original dataset, thus allowing a demonstration of clean(er) vs. dirty data. Dirty data is data with elements that either serve no purpose or interfere with the actual data processing. One main rule of computers is GIGO: Garbage In, Garbage Out – and dirty data is the electronic equivalent of rubbish.

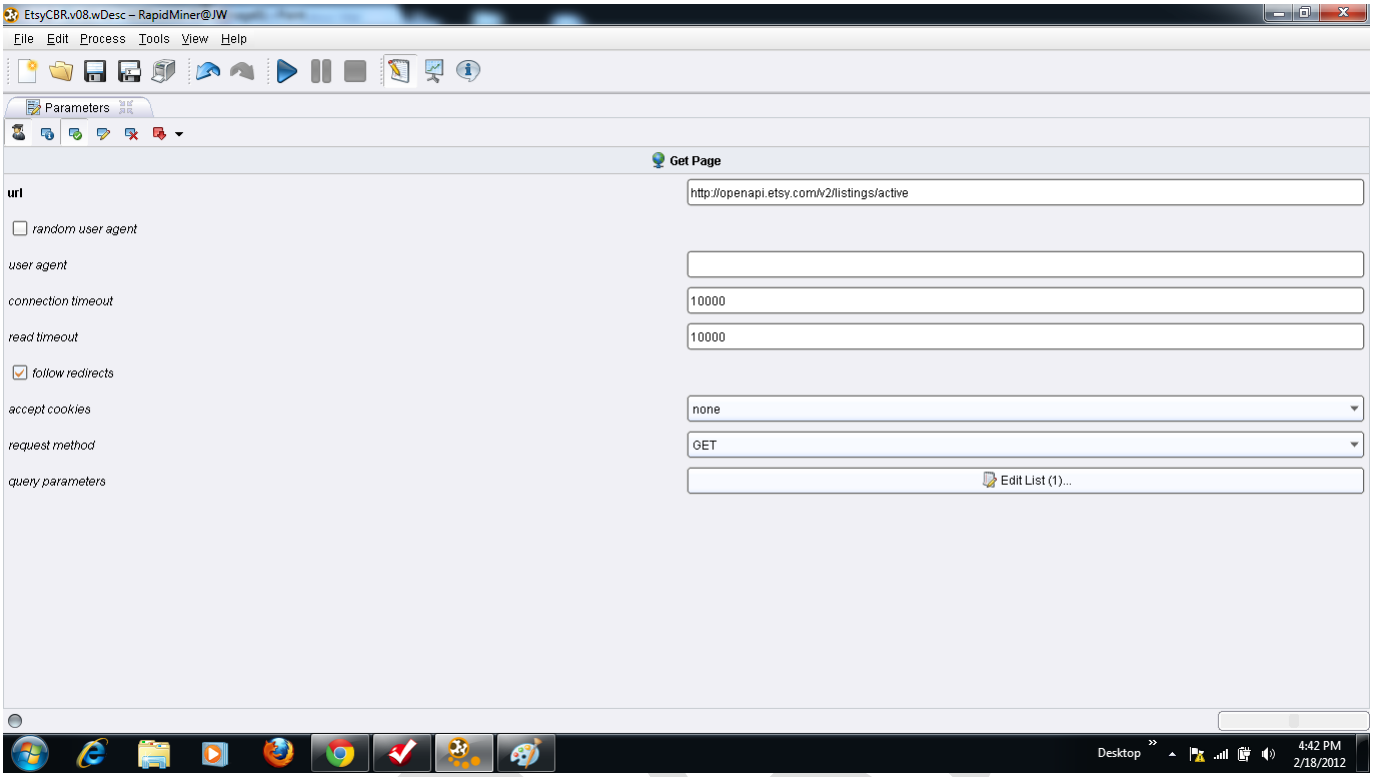


Image 2. Getting The Page. This displays the settings for web page retrieval. Most API feeds are constantly streaming data flows, with elements being added to the stream on a regular basis. Using a data set from the World Bank would be slightly different, as this data would be stored as an Excel file on your computer and then analyzed from your desktop using RapidMiner. Another option is to use RapidAnalytics, a server-deployed version of RapidMiner for large-scale organizational use.

It is important to understand how your data is structured, and how to get it into the form that you want. If you were using an Excel datasheet, then the data is already organized; a text document is much more difficult to work with, and must be re-structured in order to make use of the contents. You can estimate how difficult a data analysis process is going to be by how many steps are involved and the organization of the data: in the case of this API, the data is semi-organized, and must be converted from a web application's API to a spreadsheet – hence, it is a moderately complex task to set up.

More than anything else, it just takes patience and persistence to understand technology. If you keep hammering away at the keys long enough, I can almost promise you that you will get results. It may not be this day, or even this week, but eventually there will come a day when something will coalesce. During this time, you are also beginning to understand the fundamental structure of computers and information: mathematics and formal logic rule the day in the IT world.

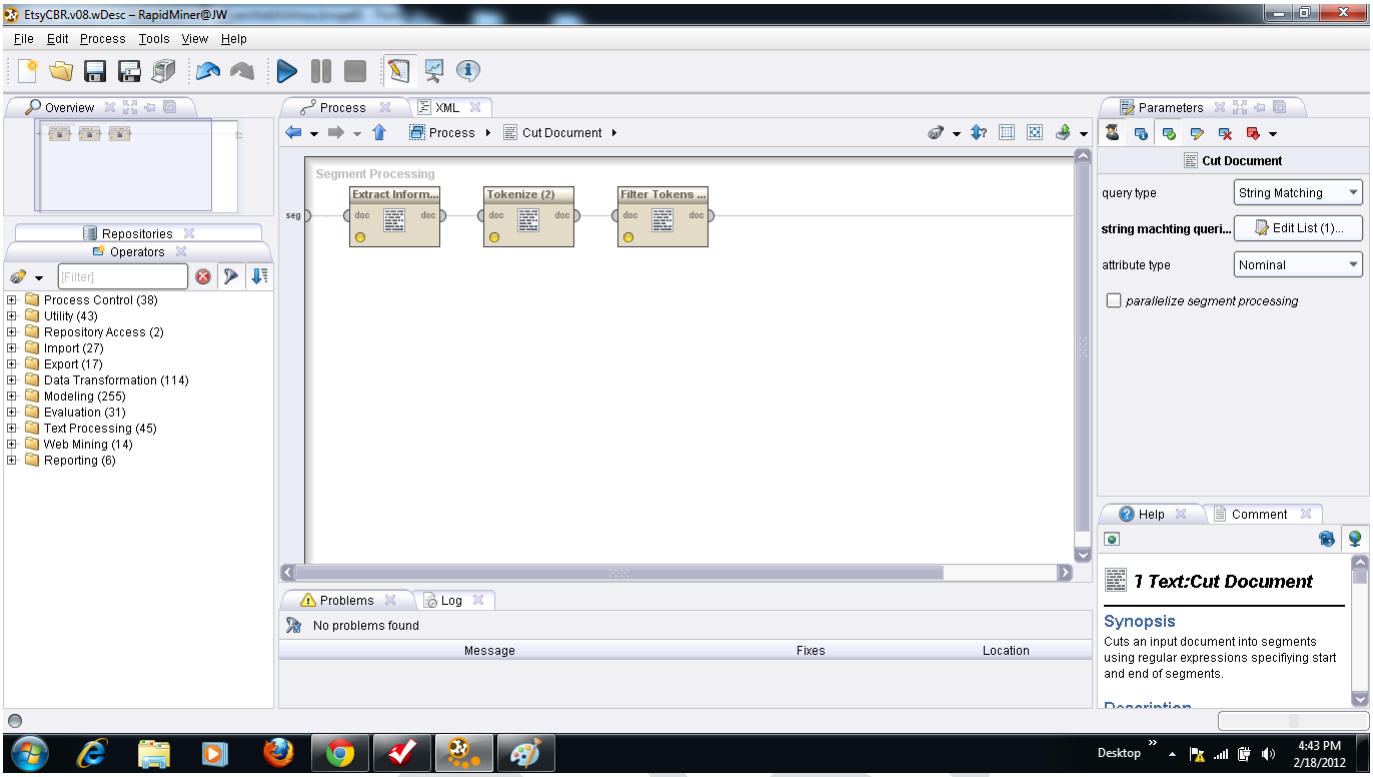


Image 3. Cutting the Document (Sub-Process). You remember how I mentioned that there was a sub-process involved with cutting the document into pieces? This is displayed here as a series of three actions that RapidMiner takes in order to break the data feed apart. If you were using a text document, these steps would be almost identical – especially the core concept of tokenization.

First the data is extracted (Extract Information), which is a fairly lengthy process to design. Every element that is wanted is extracted. The chunk of data corresponding to the element's start and end is stored in an array (think of it like a temporary holding place). Then the data is tokenized (Tokenize), which means that it is broken up further, according to the rules that you define. Then the Filter Token command discards the tokens that don't match your needs, in order to give you cleaner, more useful data. For example, what if your data set cannot have any punctuation? This is fairly common with older software, where punctuation will cause bugs and glitches in the output. By Tokenizing and then Filtering the Tokens, you can discard all punctuation, leaving you with a functional data set.

I go into more detail about each of these functions, which are really performing the brunt of the data processing. Keep in mind that if you were to break a text document into pieces using tokenizing, you may actually eliminate important words, meanings, or sentiments that should be retained. At the same time, if you are looking for similarities or keywords, then the Tokenize command will yield you a fairly precise key word index within seconds. As always, it depends on your data needs and wants.

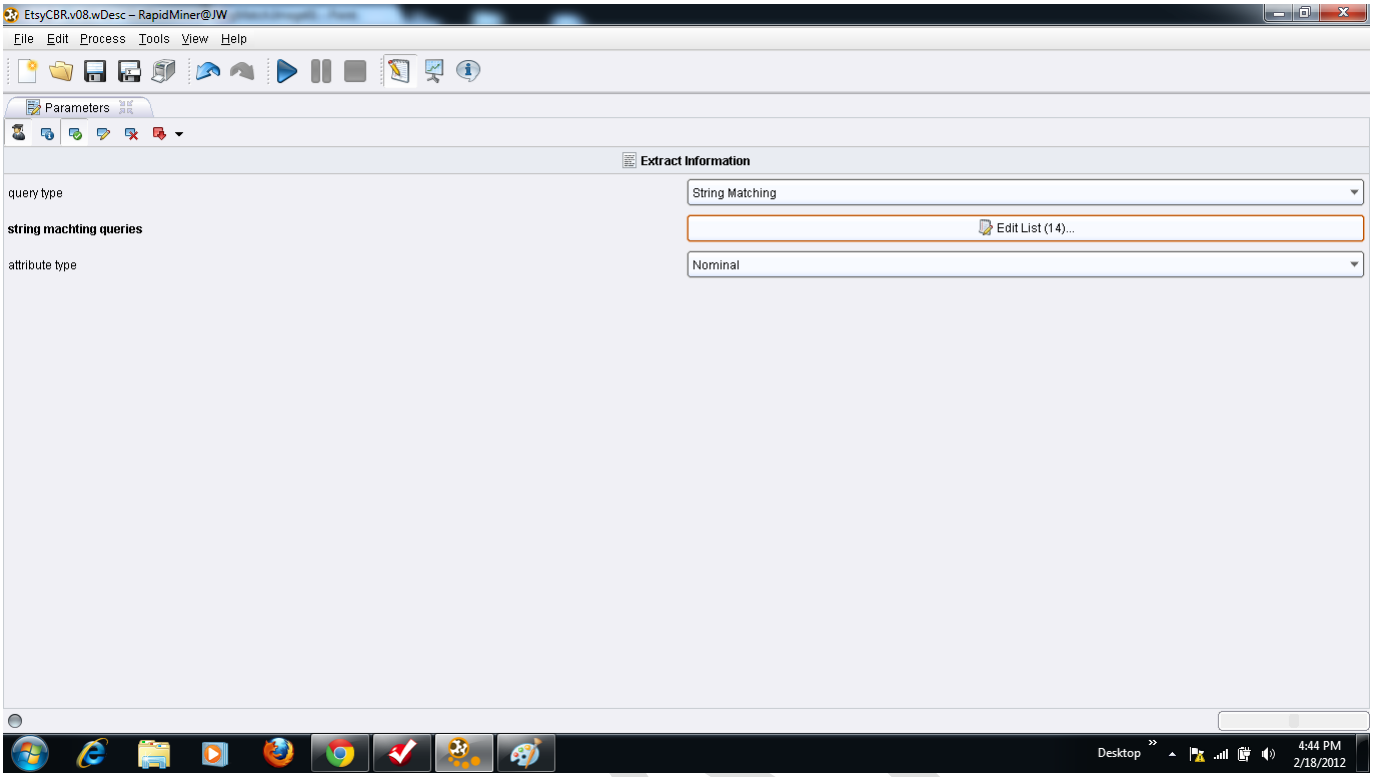


Image 4. Extract Information. This is the part of the process that will decide how to split you data into pieces. For more details, refer to Image 5 as well.

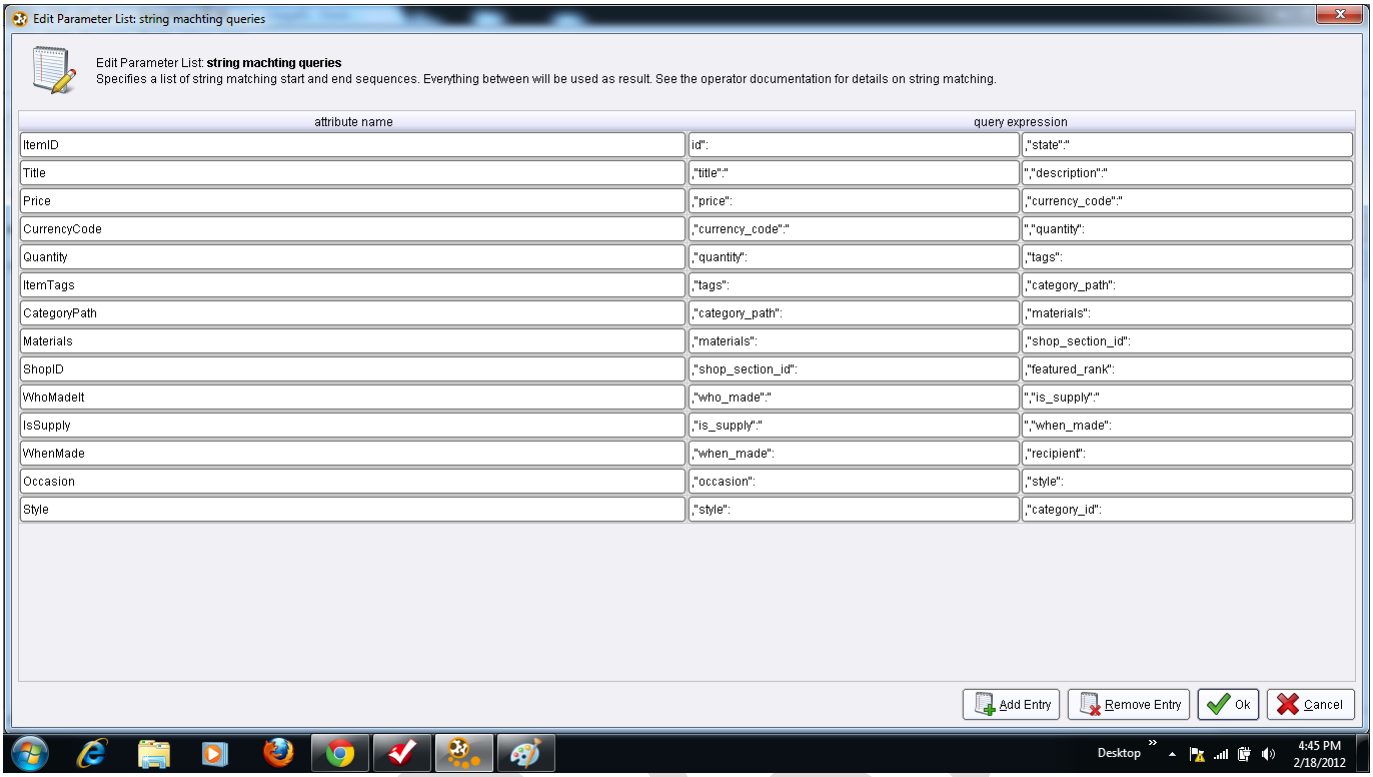


Image 5. Extract Information (Detailed View). Here you can see the method that is used to extract information from any given unorganized or partially organized data source. For instance, say your data has a field for date, but it is mashed in with all other types of data in a giant blob. By knowing how to use a command similar to Extract Information, you can find the specific element that you are looking for and pull that out. This is useful for all types of result-finding processes, and can be applied with text analytics, web scraping, data mining, and numerous other fields.

Currently, this process is set up using a String Matching algorithm. By using a more complex system of commands called 'regex', a finer control and more efficient process can sometimes be developed. It used to be that applications had to be absolutely efficient due to the limited available memory in older computers, but now there is much more wiggle room when it comes to process design. You don't need to be absolutely obsessed with precision at this point; it is much more about getting results than getting perfect results. Once you successfully get data coming out of your system, then you can refine your results. Very few people release a perfect system the first time around.

.

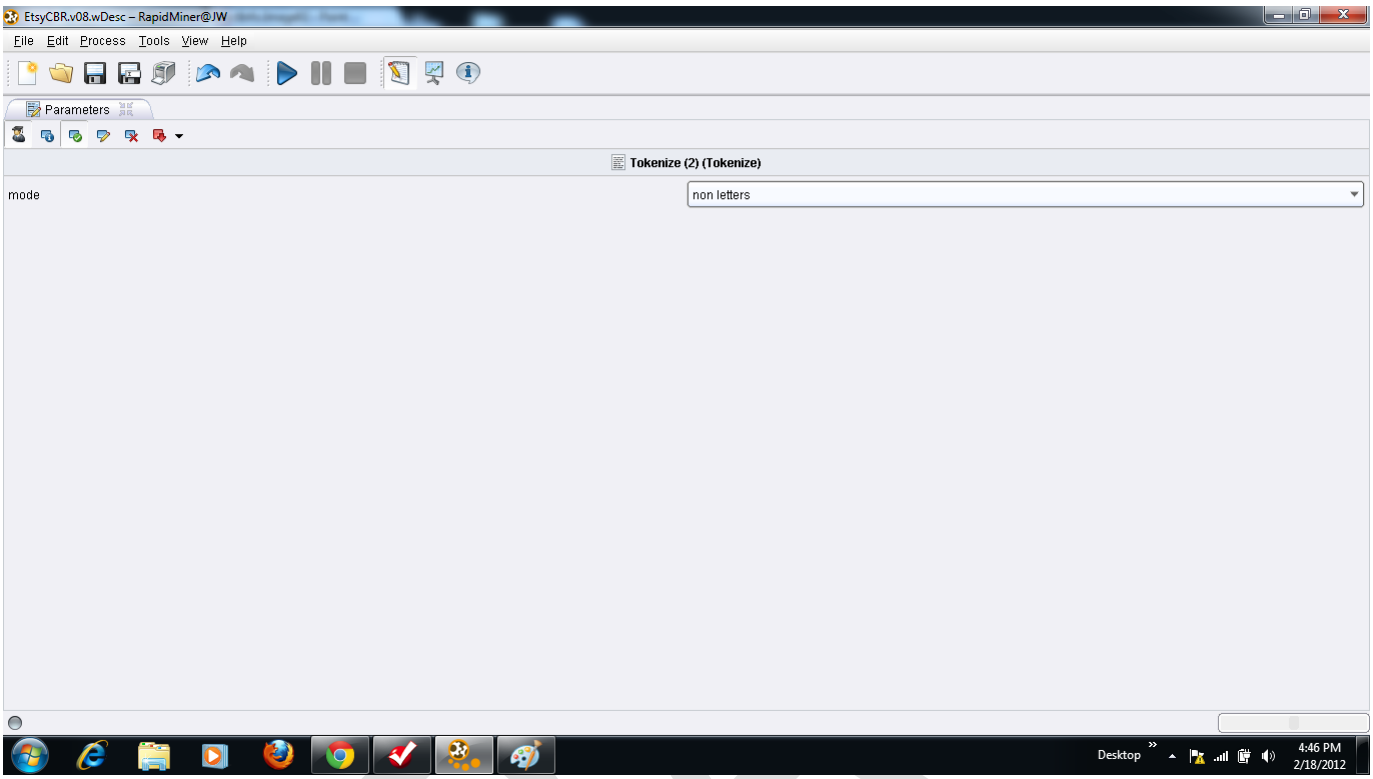


Image 6. Tokenize. This command turns all the text into tokens, which are exactly what they sound like – one token for one word. A computer doesn't care what word a token is, or the length of a token, or even if the token is in English or French. As far as the computer is concerned, there is a token in the system, and that is as far as the logic goes. Thus, tokenizing is a great way to get rid of things like non-letters – but be warned, the computer won't assist you in extracting useful or important tokens from tokens that carry little meaning.

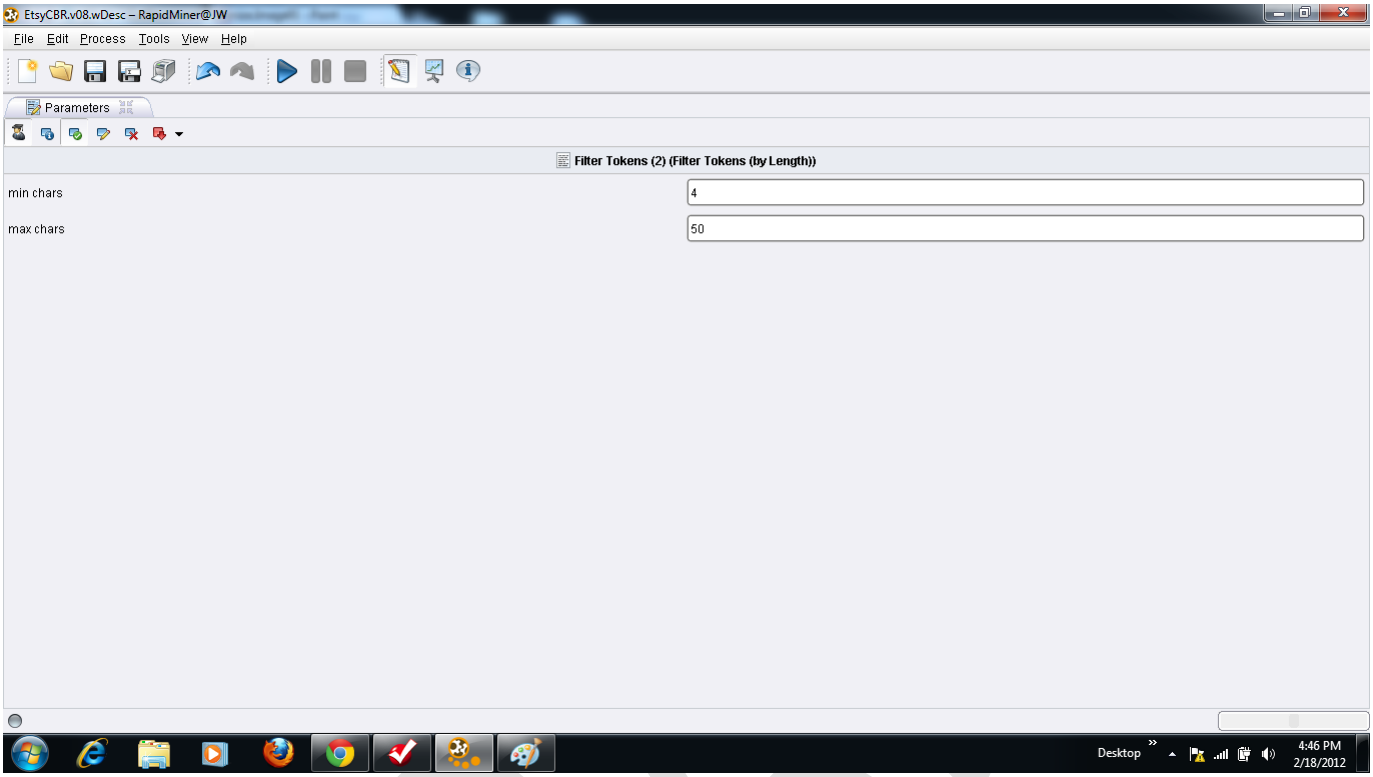


Image 7. Filtering Tokens. Filtering the tokens, in this case, is performed by the Filter Tokens command. The parameters for the token filtering are set as such: a token shorter than 4 characters (letters), and longer than 50, get cut out of the result set. Thus, the token 'the' (3 characters) gets cut out, while the token 'there' (5 characters) remains within the data set.

Filtering tokens is highly useful for separating small, semantic units from larger words. If you set the minimum character length to 3 characters, then words such as 'a', 'an', 'the', 'if', etc. all are cut out, which leaves a theoretically higher-quality key word set. As before, part of this really depends on what you are trying to achieve, what you need to keep, and what you are going to do with the results once you are done.

Don't be afraid of data and word parsing. You in no way effect the original data unless you are working from a spreadsheet or database and write the results back to the database/spreadsheet itself. This is one reason that this example is set up with CSV files: so that way you can examine the output and begin to understand how data flows and works without necessarily setting up an entire database in order to do so. As you continue to work with data, things such as parsing methods and data storage will become clearer to you.

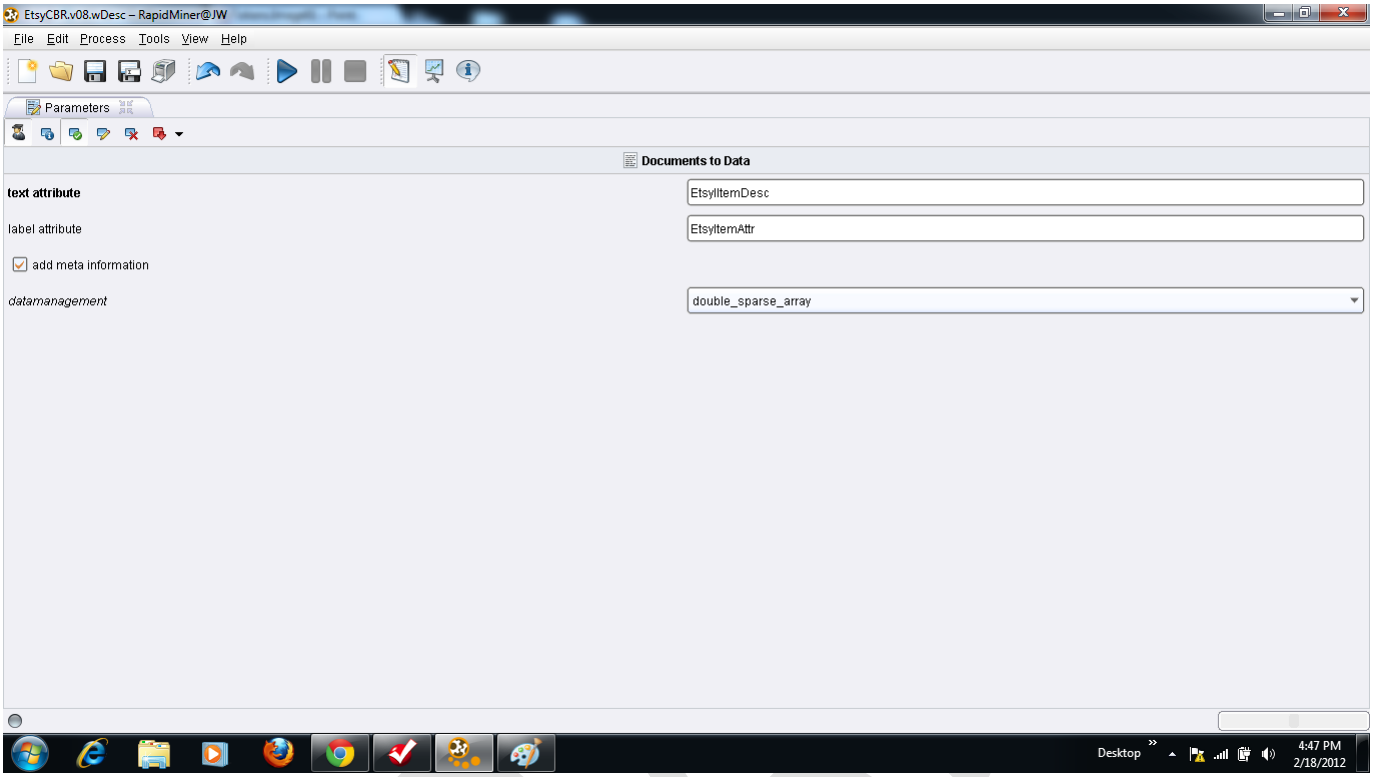


Image 8. Document to Data. This is the process that will store the output from the data process as data. When dealing with data, it is important to keep a mental note of what format your data set is in at any given time. For instance, if you are working with a document, you cannot just save to Excel – the data must be prepared first, via a routine like Documents to Data. Think of it like wrapping a gift for a friend: you need to make sure that everything will fit into the box before you deliver it.

Data in RapidMiner comes in several formats, such as Output, Data, Document, etc. These are definitely worth exploring, and every data format has its uses. There is no clear preference for one, and there shouldn't be. Just because a data set is in a numerical format doesn't make it more or less valuable than data stored in a document file. Discard preferences and prejudices against math, numbers, words, and text early on – it will be to your benefit to keep an open, flexible mind when it comes to using and manipulating data in any format.

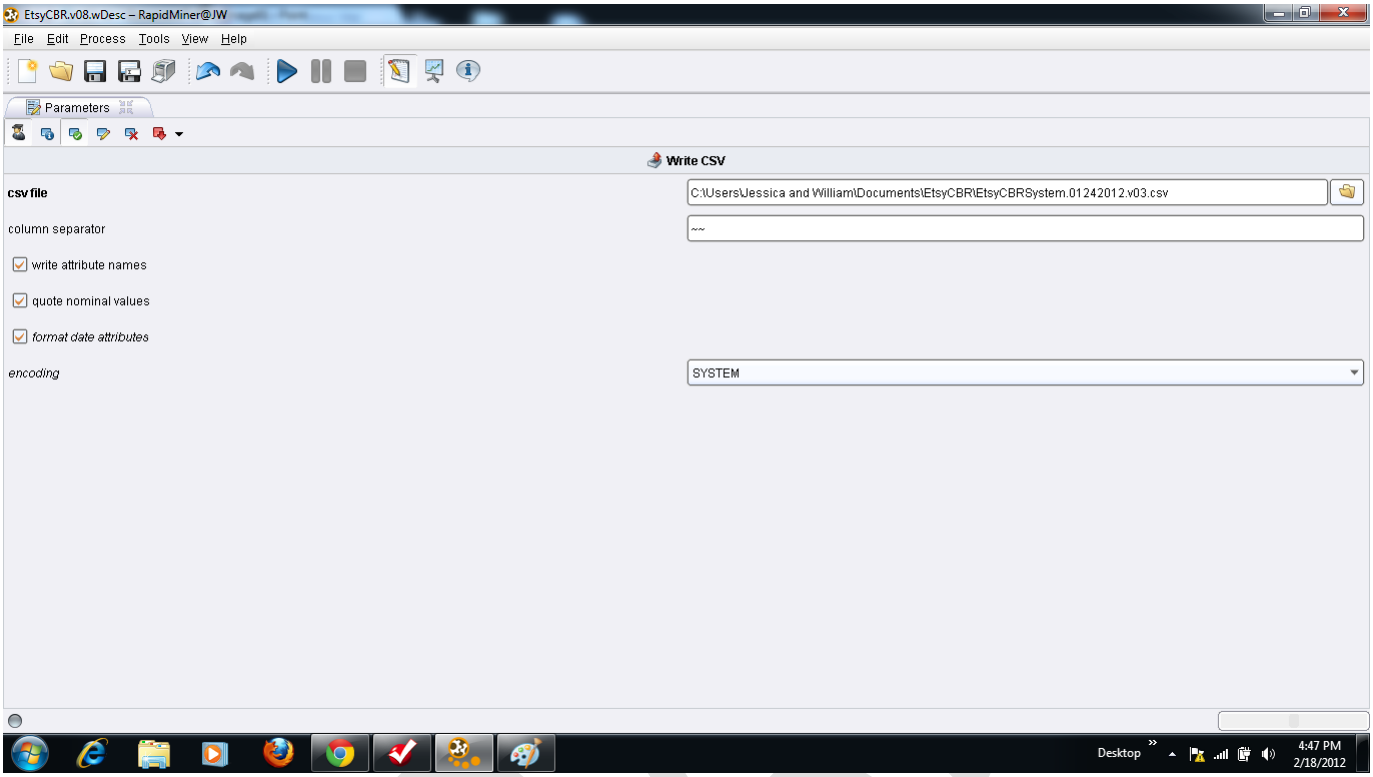


Image 9. Write CSV. This is the step that writes the CSV file to the hard drive. Other options include Excel and several different types of databases. For simplicity, the CSV format is used here, but don't feel constrained. If you want to experiment with feeding your parsed and organized data into a database, go for it.

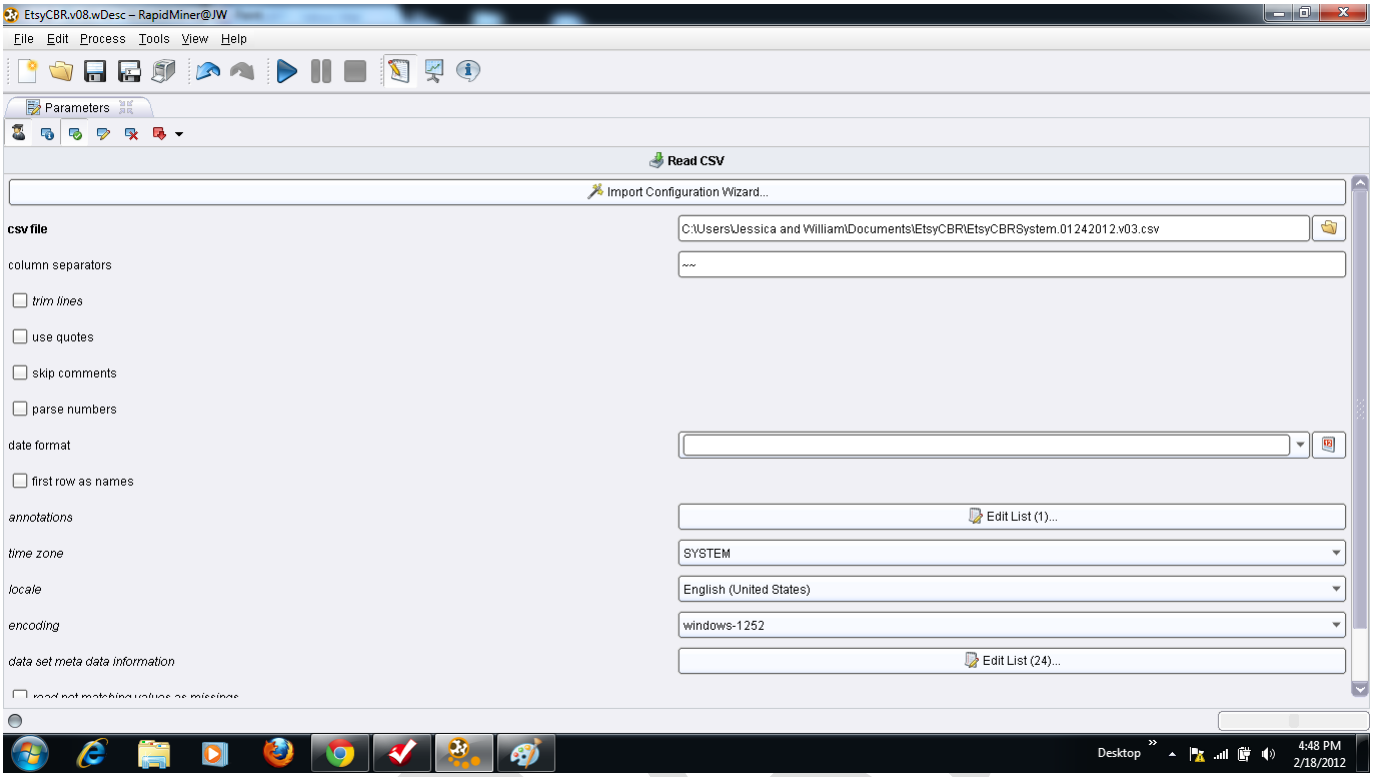


Image 10. Read CSV. This is the first step of the second process that cleans and organizes the semi-raw output, preparing it for greater analysis. There are a few approaches to data storage and manipulation, but my school of thought is that you don't want to eliminate your original data – ever. Keep in mind that the data project you are working on currently may be completely different than the data you are required to extract tomorrow, and that your data set or data mining process could be a vital feature for your future work. Save your work, and don't be afraid to refer back to it and use what you created before. Refinement and change are fundamental technical skills.

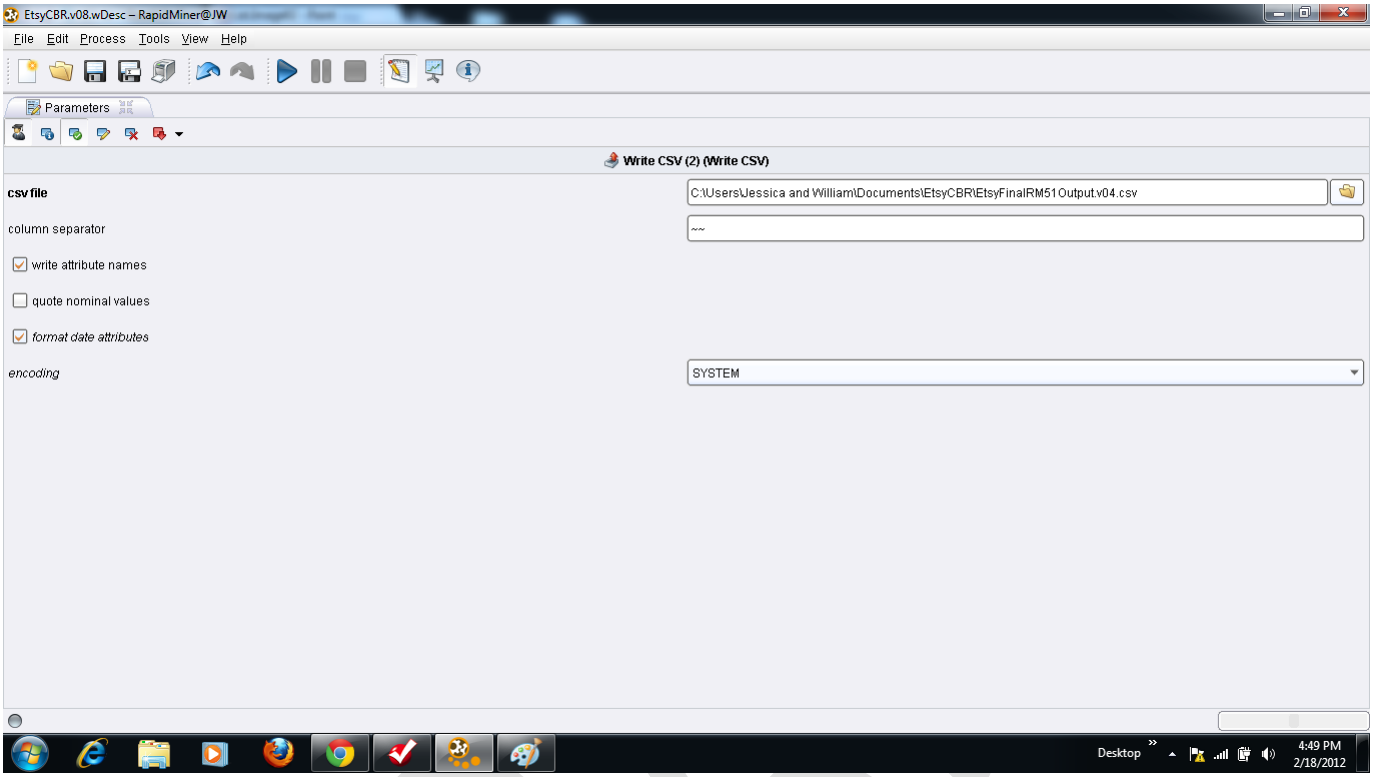


Image 11. Writing the second CSV. This shows how the refined data is saved as a different CSV file. In a database, this would be stored as another table.

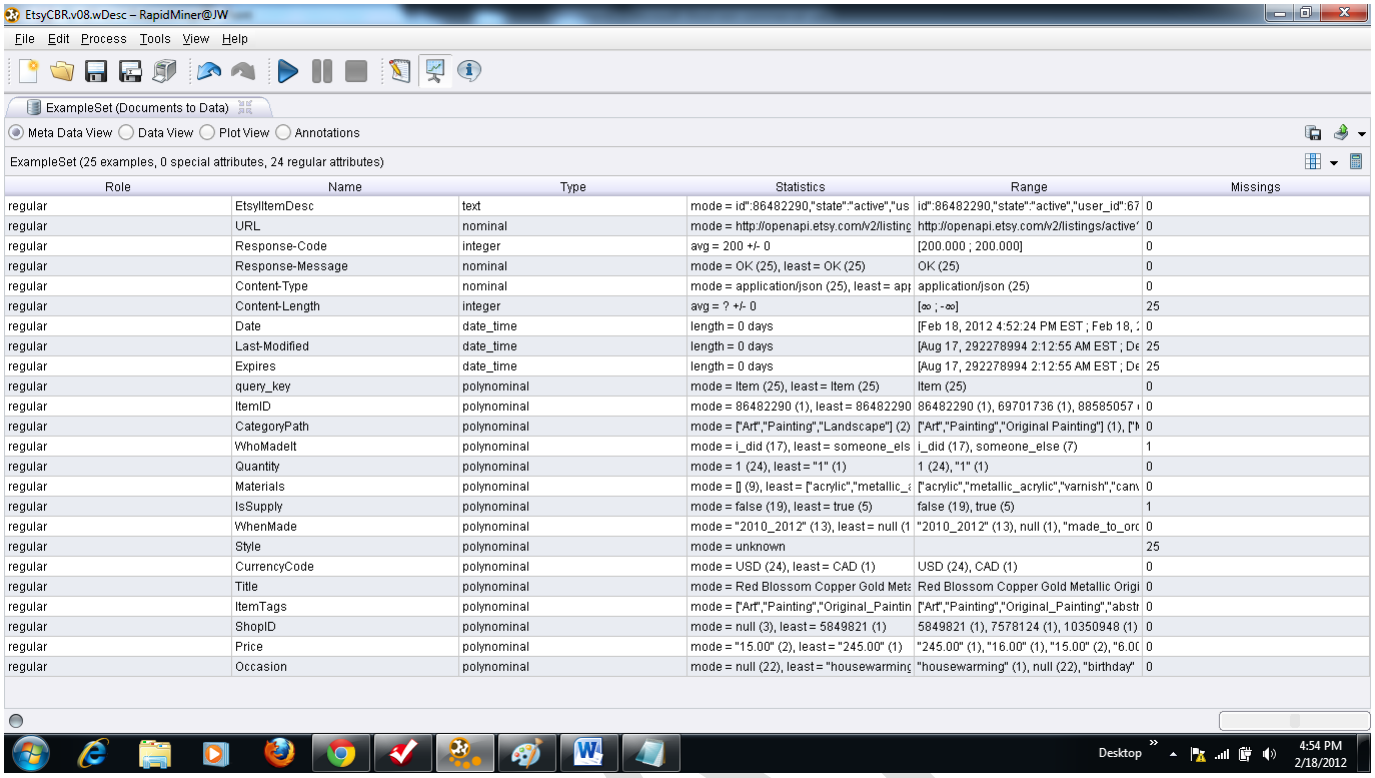


Image 12. Output from Process 1. You can see the larger size of this dataset, when compared to the output from Process 2. This would be considered the unrefined, base data set that you were working from.

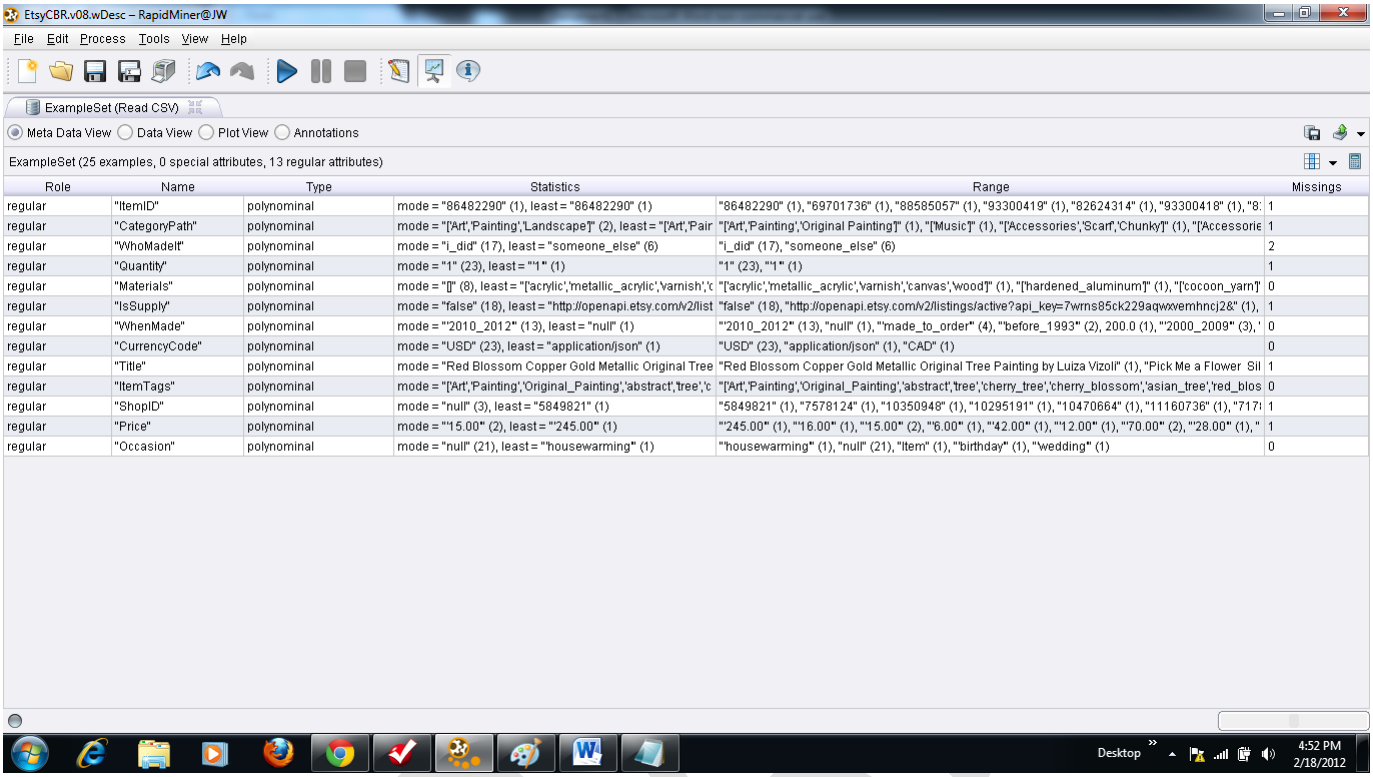
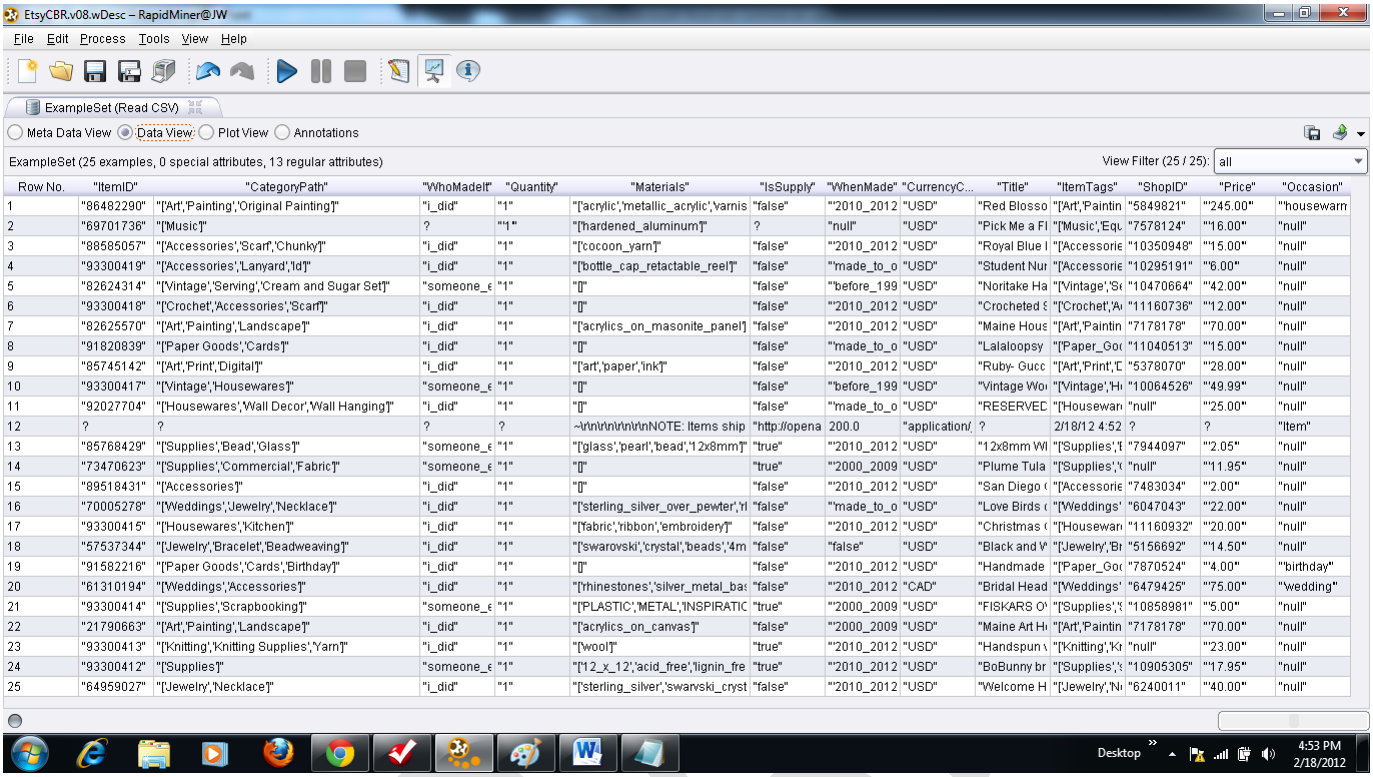


Image 13. Sample output from Process 2. This shows the output from the second process, where you can see specific categories and data types that have been parsed and stored by RapidMiner. Also included is a section on statistics, as well as the data range. These are key factors, and are demonstrated more in Image 12.



Row No.	"ItemID"	"CategoryPath"	"WhoMadeIt"	"Quantity"	"Materials"	"IsSupply"	"WhenMade"	"CurrencyC..."	"Title"	"ItemTags"	"ShopID"	"Price"	"Occasion"
1	"86482290"	"['Art','Painting','Original Painting']"	"i_did"	"1"	"['acrylic','metallic_acrylic','varnis	"false"	"'2010_2012	"USD"	"Red Blosso	"['Art','Paintin	"5849821"	"'245.00"	"housewarm
2	"69701736"	"['Music']"	?	"'1"	"['hardened_aluminum']"	?	"null"	"USD"	"Pick Me a Fl	"['Music','Equ	"7578124"	"'16.00"	"null"
3	"88585057"	"['Accessories','Scarf','Chunky']"	"i_did"	"1"	"['cocoon_yarn']"	"false"	"'2010_2012	"USD"	"Royal Blue I	"['Accessorie	"10350948"	"'15.00"	"null"
4	"93300419"	"['Accessories','Lanyard','Id']"	"i_did"	"1"	"['bottle_cap_retactable_reel']"	"false"	"'made_to_o	"USD"	"Student Nur	"['Accessorie	"10295191"	"'6.00"	"null"
5	"82624314"	"['Vintage','Serving','Cream and Sugar Set']"	"someone_e	"1"	"[]"	"false"	"'before_199	"USD"	"Noritake Ha	"['Vintage','Se	"10470664"	"'42.00"	"null"
6	"93300418"	"['Crochet','Accessories','Scarf']"	"i_did"	"1"	"[]"	"false"	"'2010_2012	"USD"	"Crocheted S	"['Crochet','Ac	"11160736"	"'12.00"	"null"
7	"82625570"	"['Art','Painting','Landscape']"	"i_did"	"1"	"['acrylics_on_masonite_panel']"	"false"	"'2010_2012	"USD"	"Maine Hous	"['Art','Paintin	"7178178"	"'70.00"	"null"
8	"91820839"	"['Paper Goods','Cards']"	"i_did"	"1"	"[]"	"false"	"'made_to_o	"USD"	"Lalaloopsy	"['Paper_Goo	"11040513"	"'15.00"	"null"
9	"85745142"	"['Art','Print','Digital']"	"i_did"	"1"	"['art','paper','ink']"	"false"	"'2010_2012	"USD"	"Ruby- Gucc	"['Art','Print','D	"5378070"	"'28.00"	"null"
10	"93300417"	"['Vintage','Housewares']"	"someone_e	"1"	"[]"	"false"	"'before_199	"USD"	"Vintage Wo	"['Vintage','Ho	"10064526"	"'49.99"	"null"
11	"92027704"	"['Housewares','Wall Decor','Wall Hanging']"	"i_did"	"1"	"[]"	"false"	"'made_to_o	"USD"	"RESERVED	"['Housewar	"null"	"'25.00"	"null"
12	?	?	?	?	"~\r\n\r\n\r\n\r\nNOTE: Items ship	"http://opena	200.0	"application/	?	2/18/12 4:52	?	?	"Item"
13	"85768429"	"['Supplies','Bead','Glass']"	"someone_e	"1"	"['glass','pearl','bead','12x8mm']"	"true"	"'2010_2012	"USD"	"12x8mm Wl	"['Supplies','B	"7944097"	"'2.05"	"null"
14	"73470623"	"['Supplies','Commercial','Fabric']"	"someone_e	"1"	"[]"	"true"	"'2000_2009	"USD"	"Plume Tula	"['Supplies','N	"null"	"'11.95"	"null"
15	"89518431"	"['Accessories']"	"i_did"	"1"	"[]"	"false"	"'2010_2012	"USD"	"San Diego (	"['Accessorie	"7483034"	"'2.00"	"null"
16	"70005278"	"['Weddings','Jewelry','Necklace']"	"i_did"	"1"	"['sterling_silver_over_pewter','rl	"false"	"'made_to_o	"USD"	"Love Birds (	"['Weddings'	"6047043"	"'22.00"	"null"
17	"93300415"	"['Housewares','Kitchen']"	"i_did"	"1"	"['fabric','ribbon','embroidery']"	"false"	"'2010_2012	"USD"	"Christmas (	"['Housewar	"11160932"	"'20.00"	"null"
18	"57537344"	"['Jewelry','Bracelet','Beadweaving']"	"i_did"	"1"	"['swarovski','crystal','beads','4m	"false"	"false"	"USD"	"Black and W	"['Jewelry','Br	"5156692"	"'14.50"	"null"
19	"91582216"	"['Paper Goods','Cards','Birthday']"	"i_did"	"1"	"[]"	"false"	"'2010_2012	"USD"	"Handmade	"['Paper_Goo	"7870524"	"'4.00"	"'birthday"
20	"61310194"	"['Weddings','Accessories']"	"i_did"	"1"	"['rhinestones','silver_metal_bas	"false"	"'2010_2012	"CAD"	"Bridal Head	"['Weddings'	"6479425"	"'75.00"	"'wedding"
21	"93300414"	"['Supplies','Scrapbooking']"	"someone_e	"1"	"['PLASTIC','METAL','INSPIRATIC	"true"	"'2000_2009	"USD"	"FISKARS O\	"['Supplies','S	"10858981"	"'5.00"	"null"
22	"21790663"	"['Art','Painting','Landscape']"	"i_did"	"1"	"['acrylics_on_canvas']"	"false"	"'2000_2009	"USD"	"Maine Art H	"['Art','Paintin	"7178178"	"'70.00"	"null"
23	"93300413"	"['Knitting','Knitting Supplies','Yarn']"	"i_did"	"1"	"['wool']"	"true"	"'2010_2012	"USD"	"Handspun \	"['Knitting','Kr	"null"	"'23.00"	"null"
24	"93300412"	"['Supplies']"	"someone_e	"1"	"['12_x_12','acid_free','lignin_fre	"true"	"'2010_2012	"USD"	"BoBunny br	"['Supplies','S	"10905305"	"'17.95"	"null"
25	"64959027"	"['Jewelry','Necklace']"	"i_did"	"1"	"['sterling_silver','swarvski_cryst	"false"	"'2010_2012	"USD"	"Welcome H	"['Jewelry','Ne	"6240011"	"'40.00"	"null"

Image 14. Data View of Output from Process 2. This shows a demonstration of the actual data pulled from an API feed. The purpose of this is showing how your data should/could look after processing. Note that there are still blanks and gaps in the data, which isn't unexpected. These could change the output or parsing, and imperfect data does create imperfect results. Knowing the limits of your data set and data process are critical.

You can also see how the data has been refined, with several data categories eliminated. This 'tidying up' is very common with data sets, and shows the critical importance of keeping your original, larger data set as well. After all, if you needed something from your larger data set, it will still be there. From a purely anecdotal perspective, I can only advise you to keep your larger data sets, even if they are dirty – you never know when it will be applicable to your work.

Appendix A. XML Output from RapidMiner 5.1 API Data Process

I decided to include the XML Output from the RapidMiner 5.1 Process from Chapter 1 for you, so you can see how the data extraction process works in real life. Most systems will allow you to examine the actual process behind your application: think of it like source code. This is the work that the system is doing on your behalf, and understanding how it works can only help when it comes to troubleshooting and innovating.

1. <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2. */// API data eliminated*
3. </list>
4. </operator>
5. <operator activated="true" class="text:cut_document" compatibility="5.1.004" expanded="true" height="60" name="Cut Document" width="90" x="179" y="30">
6. <list key="string_machting_queries">
7. <parameter key="Item" value="{"listing_.},"/>
8. </list>
9. <list key="regular_expression_queries"/>
10. <list key="regular_region_queries"/>
11. <list key="xpath_queries"/>
12. <list key="namespaces"/>
13. <list key="index_queries"/>
14. <process expanded="true" height="413" width="882">
15. <operator activated="true" class="text:extract_information" compatibility="5.1.004" expanded="true" height="60" name="Extract Information" width="90" x="45" y="30">
16. <list key="string_machting_queries">
17. <parameter key="ItemID" value="id":.,"state":""/>
18. <parameter key="Title" value=","title":".","description":""/>
19. <parameter key="Price" value=","price":.,"currency_code":""/>
20. <parameter key="CurrencyCode" value=","currency_code":".","quantity":"/>
21. <parameter key="Quantity" value=","quantity":.,"tags":"/>
22. <parameter key="ItemTags" value=","tags":.,"category_path":"/>
23. <parameter key="CategoryPath" value=","category_path":.,"materials":"/>
24. <parameter key="Materials" value=","materials":.,"shop_section_id":"/>
25. <parameter key="ShopID" value=","shop_section_id":.,"featured_rank":"/>
26. <parameter key="WhoMadeIt" value=","who_made":".","is_supply":""/>
27. <parameter key="IsSupply" value=","is_supply":".","when_made":"/>
28. <parameter key="WhenMade" value=","when_made":.,"recipient":"/>
29. <parameter key="Occasion" value=","occasion":.,"style":"/>
30. <parameter key="Style" value=","style":.,"category_id":"/>
31. </list>
32. <list key="regular_expression_queries"/>
33. <list key="regular_region_queries"/>
34. <list key="xpath_queries"/>
35. <list key="namespaces"/>
36. <list key="index_queries"/>
37. </operator>
38. <operator activated="true" class="text:tokenize" compatibility="5.1.004" expanded="true" height="60" name="Tokenize (2)" width="90" x="179" y="30"/>
39. <operator activated="true" class="text:filter_by_length" compatibility="5.1.004" expanded="true" height="60" name="Filter Tokens (2)" width="90" x="313" y="30">
40. <parameter key="max_chars" value="50"/>
41. </operator>
42. <connect from_port="segment" to_op="Extract Information" to_port="document"/>
43. <connect from_op="Extract Information" from_port="document" to_op="Tokenize (2)" to_port="document"/>
44. <connect from_op="Tokenize (2)" from_port="document" to_op="Filter Tokens (2)" to_port="document"/>

<connect from_op="Filter Tokens (2)" from_port="document" to_port="document 1"/>
<portSpacing port="source_segment" spacing="0"/>
<portSpacing port="sink_document 1" spacing="0"/>
<portSpacing port="sink_document 2" spacing="0"/>
</process>
</operator>
<operator activated="true" class="text:documents_to_data" compatibility="5.1.004" expanded="true" height="76" name="Documents to Data" width="90" x="313" y="30">
<parameter key="text_attribute" value="EtsyIItemDesc"/>
<parameter key="label_attribute" value="EtsyItemAttr"/>
</operator>
<operator activated="true" class="write_csv" compatibility="5.1.017" expanded="true" height="60" name="Write CSV" width="90" x="514" y="30">
<parameter key="csv_file" value="C:\Users\Jessica and William\Documents\EtsyCBR\EtsyCBRSystem.01242012.v03.csv"/>
<parameter key="column_separator" value="~~"/>
</operator>
<operator activated="true" class="read_csv" compatibility="5.1.017" expanded="true" height="60" name="Read CSV" width="90" x="112" y="210">
<parameter key="csv_file" value="C:\Users\Jessica and William\Documents\EtsyCBR\EtsyCBRSystem.01242012.v03.csv"/>
<parameter key="column_separators" value="~~"/>
<parameter key="use_quotes" value="false"/>
<parameter key="parse_numbers" value="false"/>
<parameter key="first_row_as_names" value="false"/>
<list key="annotations">
<parameter key="0" value="Name"/>
</list>
<parameter key="encoding" value="windows-1252"/>
<list key="data_set_meta_data_information">
<parameter key="0" value=""EtsyIItemDesc".false.polynominal.attribute"/>
<parameter key="1" value=""URL".false.binominal.attribute"/>
<parameter key="2" value=""Response-Code".false.real.attribute"/>
<parameter key="3" value=""Response-Message".false.polynominal.attribute"/>
<parameter key="4" value=""Content-Type".false.binominal.attribute"/>
<parameter key="5" value=""Content-Length".false.attribute_value.attribute"/>
<parameter key="6" value=""Date".false.binominal.attribute"/>
<parameter key="7" value=""Last-Modified".false.attribute_value.attribute"/>
<parameter key="8" value=""Expires".false.attribute_value.attribute"/>
<parameter key="9" value=""query_key".false.binominal.attribute"/>
<parameter key="10" value=""ItemID".true.polynominal.attribute"/>
<parameter key="11" value=""CategoryPath".true.polynominal.attribute"/>
<parameter key="12" value=""WhoMadeIt".true.polynominal.attribute"/>
<parameter key="13" value=""Quantity".true.polynominal.attribute"/>
<parameter key="14" value=""Materials".true.polynominal.attribute"/>
<parameter key="15" value=""IsSupply".true.polynominal.attribute"/>
<parameter key="16" value=""WhenMade".true.polynominal.attribute"/>
<parameter key="17" value=""Style".false.attribute_value.attribute"/>
<parameter key="18" value=""CurrencyCode".true.polynominal.attribute"/>
<parameter key="19" value=""Title".true.polynominal.attribute"/>
<parameter key="20" value=""ItemTags".true.polynominal.attribute"/>
<parameter key="21" value=""ShopID".true.polynominal.attribute"/>
<parameter key="22" value=""Price".true.polynominal.attribute"/>
<parameter key="23" value=""Occasion".true.polynominal.attribute"/>
</list>
<parameter key="read_not_matching_values_as_missings" value="false"/>
</operator>

```
97. <operator activated="true" class="write_csv" compatibility="5.1.017" expanded="true" height="60" name="Write CSV (2)" width="90" x="313" y="210">
98. <parameter key="csv_file" value="C:\Users\Jessica and William\Documents\EtsyCBR\EtsyFinalRM51Output.v04.csv"/>
99. <parameter key="column_separator" value="~~"/>
100.        <parameter key="quote_nominal_values" value="false"/>
101.        </operator>
102.        <connect from_op="Get Page" from_port="output" to_op="Cut Document" to_port="document"/>
103.        <connect from_op="Cut Document" from_port="documents" to_op="Documents to Data" to_port="documents 1"/>
104.        <connect from_op="Documents to Data" from_port="example set" to_op="Write CSV" to_port="input"/>
105.        <connect from_op="Write CSV" from_port="through" to_port="result 1"/>
106.        <connect from_op="Read CSV" from_port="output" to_op="Write CSV (2)" to_port="input"/>
107.        <connect from_op="Write CSV (2)" from_port="through" to_port="result 2"/>
108.        <portSpacing port="source_input 1" spacing="180"/>
109.        <portSpacing port="sink_result 1" spacing="0"/>
110.        <portSpacing port="sink_result 2" spacing="144"/>
111.        <portSpacing port="sink_result 3" spacing="0"/>
112.        </process>
113.        </operator>
114.        </process>
```

Appendix B: An Annotated Bibliography: Free Online Health Data Sources

There exists a shortage of usable data sets and public health data. Whether your interest is biomedical engineering, health informatics, data mining, or public health analysis, this annotated bibliography should contain something that will aid your search for knowledge. It is my pleasure to compile this resource for you, and I hope that you find it as useful as I have during my work as a health informaticist and data scientist. Thank you for using this research in your work, and I wish you the best on your data endeavors.

For this first edition, this bibliography is compiled alphabetically. As things progress and this work grows, it can be certain that a different shape will emerge. At the same time, the basic concept still holds true: keep it simple. If you are looking for a database, data set, visualization tool, or government health data fact, you can probably find it within one of these data sets. Please feel free to write me at http://velluminformation.com with any specific data requests or questions, and I will be happy to aid you if possible.

Cover image courtesy of twobee / FreeDigitalPhotos.net

1. caBIG Knowledge Center: https://wiki.nci.nih.gov/display/cabigkcwikis/Knowledge+Centers

The caBIG Knowledge Center is a databank hosted by the National Cancer Institute. Under its umbrella are a wiki, a forum, and a whole host of databanks. These include: caGrid Knowledge Center, Clinical Trials Management Systems Knowledge Center, Data Sharing and Intellectual Capital Knowledge Center, Imaging Knowledge Center, Molecular Analysis Tools Knowledge Center, Tissue/Biospecimen Banking and Technology Tools Knowledge Center, Vocabulary Knowledge Center, and the Development Code Repository, a Subversion server dedicated to knowledge center development code.

2. Centers for Disease Control: http://www.cdc.gov/DataStatistics/

This repository includes data and statistics via topic, including: Aging, blood disorders, cancer, chronic diseases, deaths, diabetes, genomics, growth charts, heart disease, immunizations, life expectancy, MRSA, oral health, overweight & obesity, physical inactivity, reproductive health, smoking & tobacco, STDs, vital signs, and the workplace.

3. Centers for Medicare and Medicaid Services, Data Compendium: http://www.cms.gov/DataCompendium/

“The CMS Center for Strategic Planning produces an annual CMS Data Compendium to provide key statistics about CMS programs and national health care expenditures. The CMS Data Compendium contains historic, current, and projected data on Medicare enrollment and Medicaid recipients, expenditures, and utilization. Data pertaining to budget, administrative and operating costs, individual income, financing, and health care providers and suppliers are also included. National health expenditure data not specific to the Medicare or Medicaid programs is also included making the CMS Data Compendium one of the most comprehensive sources of information available on U.S. health care finance. This CMS report is published annually in electronic form and is available for each year from 2002 through present.”

4. Community Health Profile: National Aggregate of Urban Indian Health Organization Service Areas, December 2011: http://www.uihi.org/wp-content/uploads/2011/12/Combined-UIHO-CHP_Final.pdf

This report contains statistical data for the Urban Indian Health Institute’s research: topics include sociodemographics, mortality, access to care, alcohol use, and environmental, heart, mental, and maternal/child health. Compiled from the national service areas located within the USA.

5. Data.ed.gov: http://data.ed.gov/

Includes data tools and data sets: for example, Fiscal data for public schools and universities, common data core sets, educational progress and primary/postsecondary data. Data sets include legal data, Federal resources, and trends in science and mathematics for students. Data sets are in a variety of formats, XML, CSV, and XLS.

6. Data.gov: http://www.data.gov/health

"You've found a public resource designed to bring together high-value datasets, tools, and applications using data about health and health care to support your need for better knowledge and to help you to solve problems. These datasets and tools have been gathered from agencies across the Federal government with the goal of improving health for all Americans. Check back frequently because the site will be updated as more datasets and tools become available"

Key elements include a massive index of health data sets: Medicare, geographic data, medical record system adoption, child welfare, and assisted reproduction data. There is a health apps repository/demo site, and a small collection of other data sources that bears looking at, especially for 1. California's health data, and 2. The Gallup Poll Well-Being Index.

7. Educational Data Partnership, California's K-12 Schools: http://www.ed-data.k12.ca.us/Pages/Home.aspx

Data for all of California's public school system, by State, County, District, and school. Also includes reports, teacher salaries, and data about charter schools.

8. FastStats A to Z: http://www.cdc.gov/nchs/fastats/default.htm

FastStats has data for any illness or major life complication that could arise for a citizen of the USA. A small sample includes: American Indian or Alaskan Native health, assault/homicide, cancer, deaths/mortality, emergency department visits, immunizations, kidney disease, life expectancy, marriage, Mexican American health, obesity/overweight, pertussis, smoking, and teen pregnancy. If it is a life-changing event, chances are good that FastStats has at least basic data for it.

9. Federal Government IT Dashboard: http://www.itdashboard.gov/

"The IT Dashboard is a website enabling federal agencies, industry, the general public and other stakeholders to view details of federal information technology investments. The purpose of the Dashboard is to provide information on the effectiveness of government IT programs and to support decisions regarding the investment and management of resources. The Dashboard is now being used by the Administration and Congress to make budget and policy decisions.

Importantly, there are analysis tools and data feeds, not quite a data set. Also, the source code is available for the IT Dashboard.

10. Health and Human Services Open Data Initiative: http://www.hhs.gov/open/

Includes details for mHealth Initiative, Startup America, and health data competitions. Also includes data about executive orders and records and reports.

11. Health Indicators Warehouse: http://www.healthindicators.gov/

The Health Indicators Warehouse has data sets sorted by topic, geography, and initiative. Example data sets include: Chronic Diseases, Disabilities, Health Care, County data, Community Health Data Indicators, and CMS Community Indicators. Also, data sets are available for all 50 states and Washington, D.C.

12. HealthyPeople 2020: http://www.healthypeople.gov/2020/default.aspx

The Healthy People 2020 Initiative is dedicated to creating a health environment for everyone, and contains data and publications that strive to meet this goal. It has a specific focus on health disparities and prevention efforts.

13. Justice Department's Open Data Initiative: http://www.justice.gov/open/data.html

”Publishing high-value datasets that increase accountability and responsiveness improve public knowledge of the Department of Justice and our operations, create economic opportunity, and respond to need and demands of the public are a core component of our efforts to fulfill The Open Government Directive”

Data sets available include jail data for numerous years, antitrust cases, jail census data, law enforcement data, forensic unit funding, state and Federal correctional facility data, Chapter 7 filing, Freedom of Information filings, hate crime statistics, and prosecutor data.

14. Many Eyes: http://www-958.ibm.com/software/data/cognos/manyeyes/

Donated data sets, combined with an information visualization application, creates real-time displays from an almost endless supply of data. Everything from average Canadian household expenses, to London’s air quality, to Kobe Bryant’s game scoring, and quite a bit in between. Also, the application is relatively simple to use, which means that any given data set can be visualized with little effort.

15. Massachusetts Open Data Initiative, Data Catalog: https://wiki.state.ma.us/confluence/display/data/Data+Catalog

A huge repository of open data sets from the state of Massachusetts: economic, education, geography, health, population, public safety, and technology are all covered, as well as quite a few other subjects.

16. National Cancer Institute: http://www.cancer.gov/statistics/tools

Statistical tools and data: SEER data, SEER*Stat software, health disparities calculator, Medicare-linked database, and analytic software. Also includes a bank of statistical methods for cancer, cancer survival, and geographic information systems.

17. National Center for Health Statistics: http://www.cdc.gov/nchs/

“Welcome to the National Center for Health Statistics' website, a rich source of information about America's health. As the Nation's principal health statistics agency, we compile statistical information to guide actions and policies to improve the health of our people. We are a unique public resource for health information - a critical element of public health and health policy.”Data covers: diseases, health care and coverage, injuries, life stages, populations, lifestyle factors, and more.

18. New York City’s Open Data Initiative: http://nycopendata.socrata.com/

Open data sets for everything from subway data to open-access WiFi networks, park maps, SAT scores, and filming locations. Too much of a hodge-podge of data sets to really define – besides the key element that everything is related to New York, there is no strict boundary or catalog.

19. Open Data Initiative: http://www.opendatainitiative.org/

“The Open Data Initiative is a Web 2.0 site for disseminating public data.”Includes visualize data sets for suburb safety, Australian criminology tracking, and the Saudi Arabian census. May bear further watching, or may be transitory.

20. Open Government Data Initiative, The: http://ogdisdk.cloudapp.net/

“The Open Government Data Initiative (OGDI) is an initiative led by Microsoft Public Sector Developer Evangelism team. OGDI uses the Windows Azure Platform to make it easier to publish and use a wide variety of public data from government agencies. OGDI is also a free, open source 'starter kit' with code that can be used to publish data on the Internet in a Web-friendly format with easy-to-use, open API's. OGDI-based web API's can be accessed from a variety of client technologies such as Silverlight, Flash, JavaScript, PHP, Python, Ruby, mapping web sites, etc.”

Hosted by Microsoft’s Cloud App servers, this data initiative displays visualized data sets and has a section for data developers as well.

21. Regulations.gov: http://www.regulations.gov/#!home

Database and compendium of government regulations and laws.

22. VitalStats: http://www.cdc.gov/nchs/VitalStats.htm

VitalStats includes data sets for: births, deaths, perinatal mortality, and other public use data files related to vital statistics and their usage in the USA.

23. World Bank Data: http://data.worldbank.org/

This is the motherload of all data banks. Provides access to over 7,000 indicators for global statistics, including economic, health, education, and environmental; by country, year, and topic. Also has a microdata library.