PharmaSUG 2025 - Paper SD-143

Low-Code Solutioning in SAS Viya for Automated Clinical Data Quality, Decisioning and Harmonization

Mary Dolegowski and Scott McClain, SAS

ABSTRACT

Fact checking clinical data quality is a ubiquitous need in drug trials. It's typically a very manual, code-heavy, time-consuming process. Complications increase due to multiple data vendors. The pharma customer demand is for a reduction in human error and to speed up data processing. Our SAS objective was to automate quality review using statistical models, language models and business rules to reduce inaccuracies and time. A full application interface was created to allow human review at critical points in decision making and data harmonization. The product result is an automated end-to-end, low-code pipeline that reduces human-in-the-loop manual review with Viya out-of-box capabilities, supporting better time-to-registration for drug development.

INTRODUCTION

There is nothing more in demand than the ability to harmonize expensive, high-quality research and patient data for health and life science. Solving for this bottleneck supports faster, better answers from high quality harmonization of multiple data streams. Speed-to-answers here speaks for itself.

The challenge is that incoming data frequently does not adhere to a common data form (model format) and manual review of this data from multiple contract research organizations (CROs) is time consuming, inefficient, and prone to error. The customer responsible for overseeing research trials and the incoming data via CROs encounters data that needs to be reviewed, often manually, prior to being incorporated into enterprise data storage. They examine the data quality and naming conventions multiple times as data files come into programming environments. The current state is a process of people interacting directly with spreadsheet style data forms that are not consistent through time and do not match across CROs. This is even before final checks are processed through quality assurance which adds labor and points of failure in the overall review process due to the manual nature of the review.

THE PROCESS

Figure 1: Process Flow OverviewIt is important to note that this workflow intentionally relied on the GUI interfaces and out of the box capabilities, in SAS Viya, to provide ease of access. However, the processing pipeline has low-code control layers allowing modifications and updates to incorporate more levels of complexity. Figure 1 outlines the automation process we built.

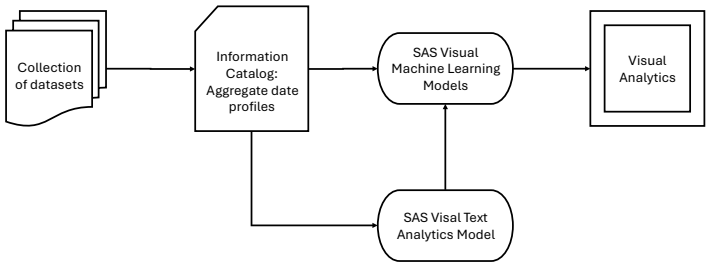


Figure 1: Process Flow Overview

We started by collecting all the available data sets into a single library and then feeding them into SAS Information Catalog to automate the profiling process. Once all of the data has been aggregated it is fed into two separate models. The first uses SAS Visual Text Anaylics (VTA) to group and connect like variable names and possible misspellings. Next, the VTA model, along with the aggregate data profiles from Infromation Catalog, are fed into Model Studio to create a new machine learning model. The final step places all of the above results in a dashboard to allow for easy comparison of historical aggregate information, new aggregate information, and the results from the machine learning models to create a one stop show for scoring the probability of inaccuracies and assigning matched data variables to known standards.

DATA CURATION

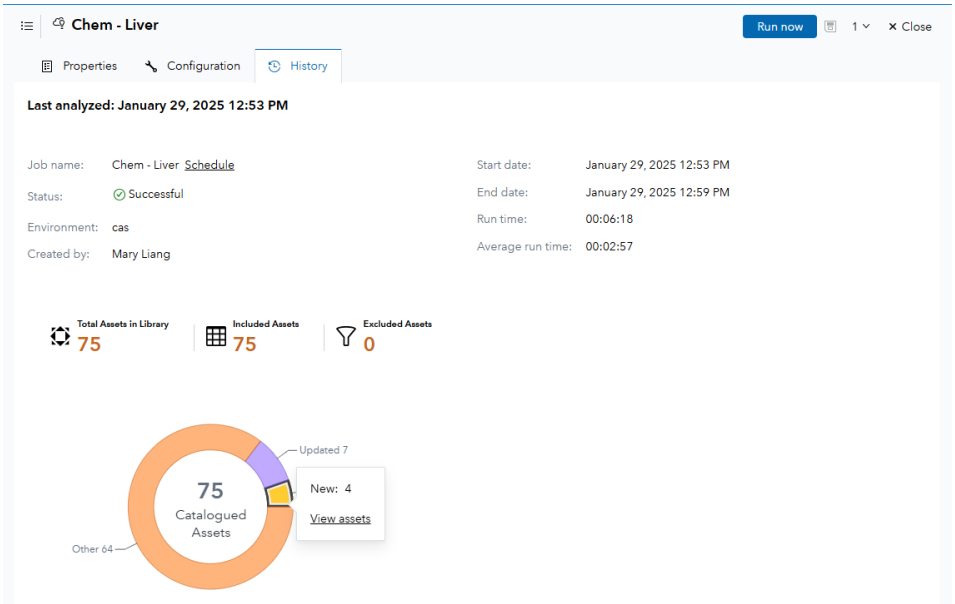
The first step in this overall process is the curation of the profiled data. This is completed using SAS Information Catalog and its ability to create bots that can profile entire libraries at the click of a button or based on a configurable schedule. This process leads to a smaller and more manageable dataset based on the aggregate of all the individual table profiles.

A cornerstone of the process is leveraging the law of large numbers. Due to requirements of processing many data sets, the ability to generate accurate profiles using numerical averages takes advantage of identifying outliers and having a base set of validated data against which to score new data.

INFORMATION CATALOG

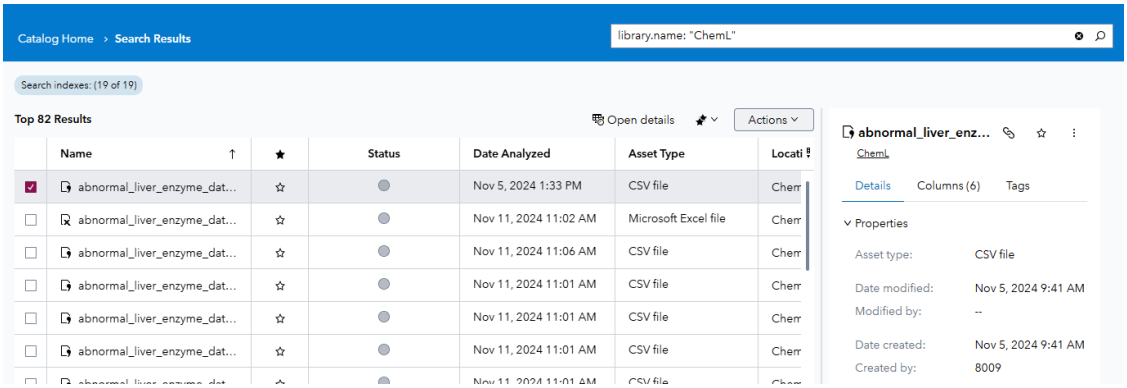
The collection of datasets to be profiled are all stored into a central location (in this case a folder on a file share) and linked to a library where they can be referenced within the Information Catalog. From there, data can be profiled two ways: as indivdual dataset or as part of a library by using bots. Bots allow for *ad hoc* auto-profiling, as shown in the top right corner of Display 1. Additionally, bot-generated auto profiling can be scheduled by clicking the schedule link next to the jobs name. This will take the user into SAS's Environment Manager where these jobs can be tracked and modified. One of the advantages of using the Information Catalog is that profiling runs as a background job, allowing users to continue working uninterrupted.

The auto-profiling bots also generate the tracking information displayed in Display 1. This includes the timestamp of the last library profiling run, its duration, the number of datasets profiled, and a breakdown of new, updated, and unchanged datasets. An interactive donut chart further enhances usability, allowing users to quickly explore any new or modified datasets.



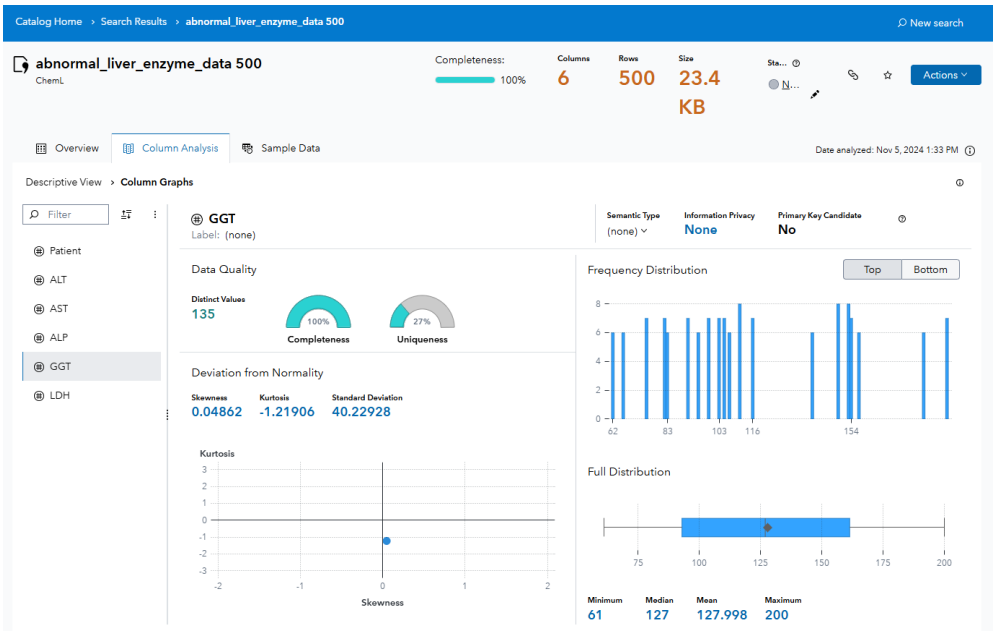
Display 1: Information Catalog - History

Display 2 presents all of the profiled data within the library. Notably, that Information Catalog is capable of profiling various files types, not only SAS file types but also CSV, Excel files, and parquet. Importantly, the data does not need to be loaded into memory (referred to as Cloud Analytic Services, or CAS) in order to be profiled. It simply needs to be accessible through the library. This provides a high degree of flexibility in terms of the data sources the library can reference.



Display 2: Information Catalog - Profiled Library

Display 3 showcases one of the interactive views available for exploring the profiled data. This view is tailored to the specific type of variable being examined, meaning each variable type will present a slightly different visualization or set of metrics. The data behind this view is the same profiled data that will be used as input for model development.



Display 3: Information Catalog - Profiled Dataset

SAS INTERNAL APIs

Once the data has been profiled, the most efficient way to retrieve the aggregated dataset is by using SAS's internal APIs. A comprehensive list of available APIs can be found at https://developer.sas.com. Program 1 provides example code demonstrating how to extract profile data from the Information Catalog for a specific library. After extraction, optional data management and cleansing steps can be applied to ensure the resulting dataset meets the format and structure requirements for use in Visual Text Analytics (VTA) and Machine Learning pipelines in Model Studio. While the overall solution is designed to be low-code, there are still a few necessary coding steps to integrate and connect key components of the workflow.

```
proc cas;

      data_file.level = "dataDictionaryAndProfile";
      data_file.prefix = "catalog";
      data_file.dateTimeStampSuffix = true;
      data_file.serverName = "cas-shared-default";
      data_file.caslibName = "&base_lib.";

      final = casl2json(data_file);

      file outfile "&output_file.";
      print final;

run;

filename d_in '&output_file';

filename resp TEMP;

proc http
      url="&baseurl./catalog/instances/?filter=and(contains(resourceId,&base_lib.),contai
ns(type,casTable))%nrstr(&limit)=500"
      method = 'POST'
      in = d_in
```

```
      out=resp
      OAUTH_BEARER = SAS_SERVICES
      CT = "application/vnd.sas.metadata.instance.upload.request+json";
      headers
            "Accept"="application/vnd.sas.metadata.instance.upload.request+json,
application/json, application/vnd.sas.error+json";

run;
```

Program 1: SAS Code - API Call to Information Catalog

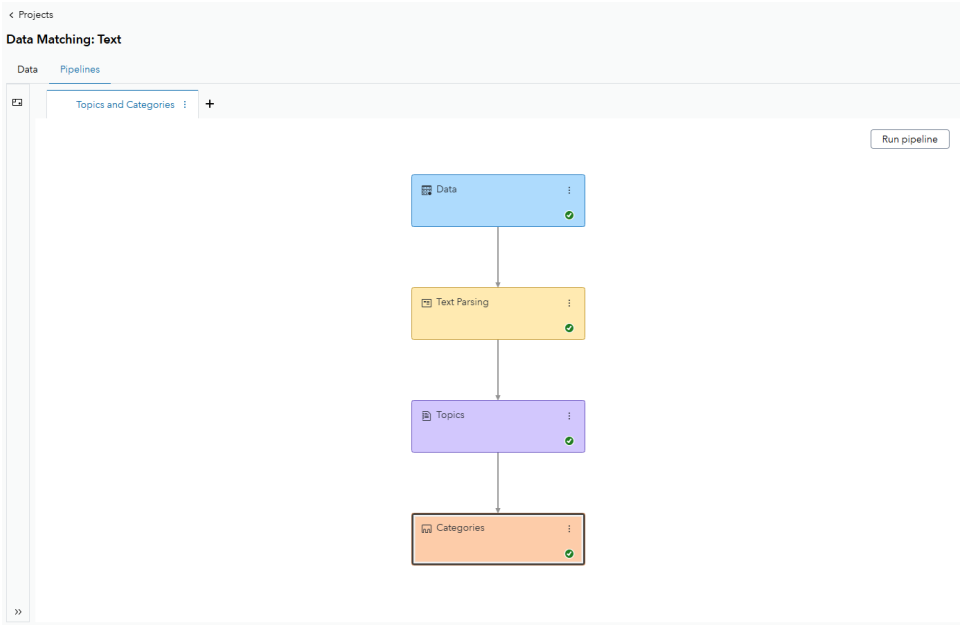
MODEL BUILDING

The next section takes the data and applies two different models to predict which variable matches with the variables in the standard we are trying to convert the dataset into. The first model leverages text analytics to group similar terms and connect them to the standard variable names. The second model builds on the results of the first by incorporating those predictions along with profiled data to generate a more robust prediction. This approach combines insights from the data's statistical profile and the semantic information embedded in the variable names themselves. In some cases, datasets include placeholder variable names such as var, simulating scenarios where variables are either misnamed or entirely unlabeled.

Both models were developed using SAS Model Studio, a graphical user interface (GUI) designed to support the construction of modeling pipelines. The VTA pipeline includes nodes for text analytics tasks such as text parsing and categorization. Meanwhile, the machine learning pipelines allow for the comparison of various modeling techniques, enabling the selection of the most effective algorithm. The following sections will provide a detailed overview of how these tools and techniques were implemented within Model Studio.

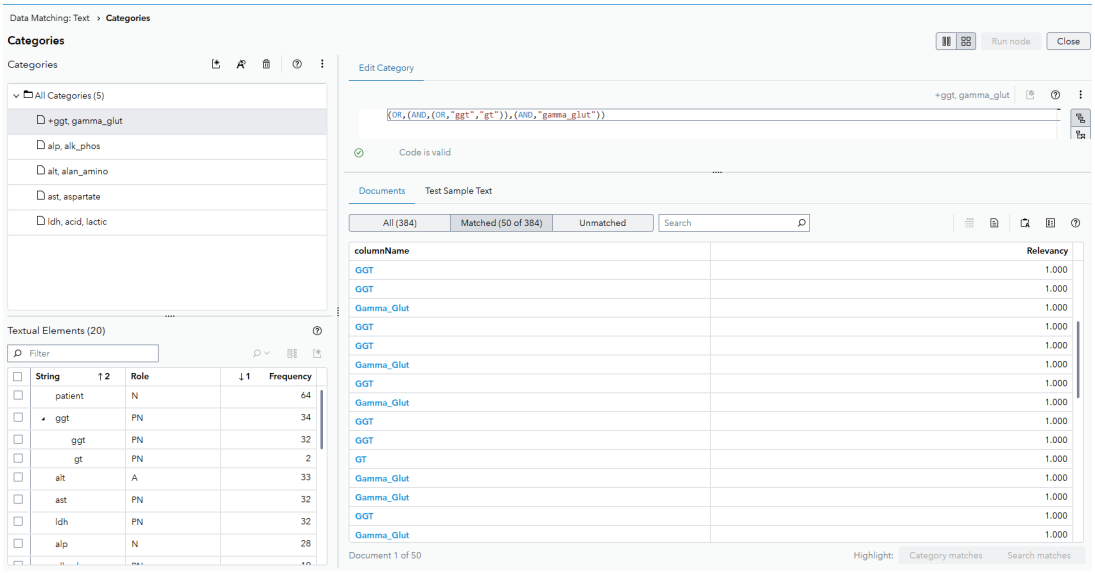
VISUAL TEXT ANALYTICS

Display 4 illustrates the default pipeline provided in VTA. Before this step, the only prerequisites are that the dataset is made available in CAS (part of the data management process outlined in Program 1) and that a text field is assigned; in this case, the variable name field from the dataset. Within the Text Parsing node, there is an option to enable misspelling detection. This feature allows similar terms (such as ‘gt’ and ‘ggt’) to be grouped automatically, eliminating the need for manual intervention by the user. The Topics node then identifies and organizes related concepts into thematic groupings. Finally, the Categories node consolidates these groupings, creating a structured representation of the underlying variable associations.



Display 4: Visual Text Analytics - Pipeline

Display 5 shows the interactive configuration options within the Categories node, along with the groupings that VTA was able to create by identifying similar terms. These grouped terms can then be mapped back to the standard. From this mapping, score code is generated. The score code outputs the probability that a given variable matches a corresponding variable in the standard.

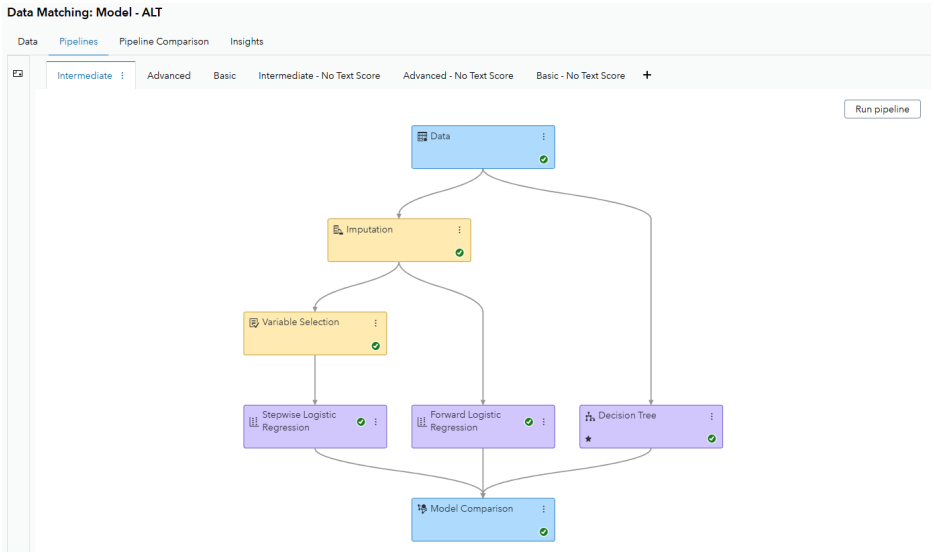


Display 5: Visual Text Analytics – Categories

MODEL STUDION - MACHINE LEARNING

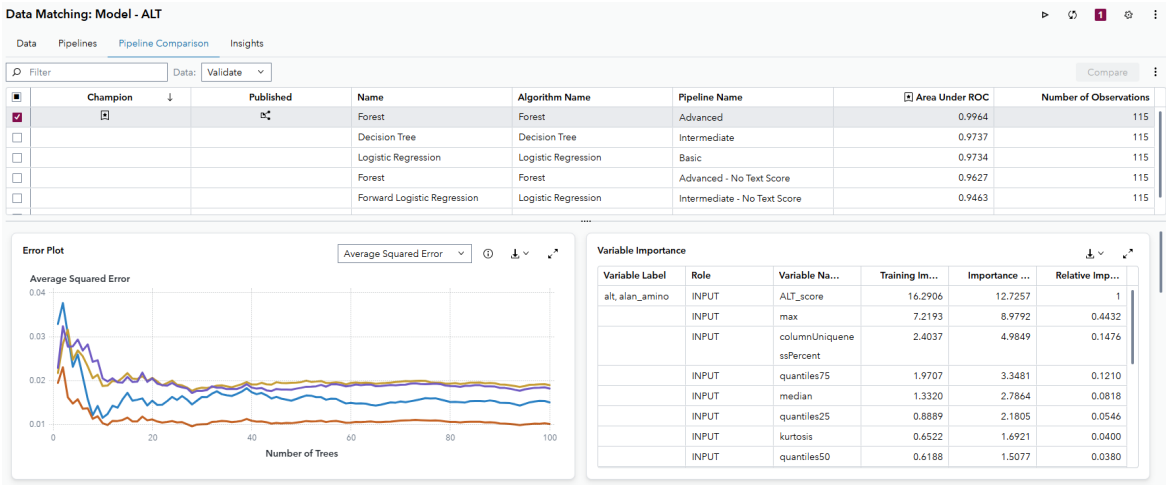
The VTA-scored data, combined with the aggregated profiled data, is loaded into Model Studio. Depending on the scoring method used, this integration may not require any additional coding. Display

shows a comparison of six different modeling pipelines. The Intermediate, Advanced, and Basic pipelines all use out-of-the-box configurations. The remaining three pipelines, Intermediate – No Text Score, Advanced - No Text Score, and Basic - No Text Score, also use the out-of-the-box pipelines, but with an added variable selection node to exclude the score generated by the VTA model. This dual setup allows for a direct comparison of model performance, helping to evaluate the impact and utility of including the VTA-generated text score in the modeling process.



Display 6: Model Studio – Machine Learning - Pipelines

Display 7 shows the champion model from each pipeline, along with the final champion model from the entire project. In this case, incorporating the text score results in a better-fitting model. This pipeline generation process must be repeated for each variable in the standard. However, the project is designed to be flexible, allowing for model updates as the datasets evolve and expand over time.



Display 7: Model Studio – Machine Learning - Pipeline Comparison

There are several methods for extracting the final model score code. One option is to add a score code node to the pipeline after the champion model, which applies the score code to a dataset. Alternatively, the champion model in the pipeline comparison can be right-clicked to download the model score code, or there is an option to published and track the model within SAS Model Manager. All of these options offers different benefits, depending on the stage of development and the user's needs within the process.

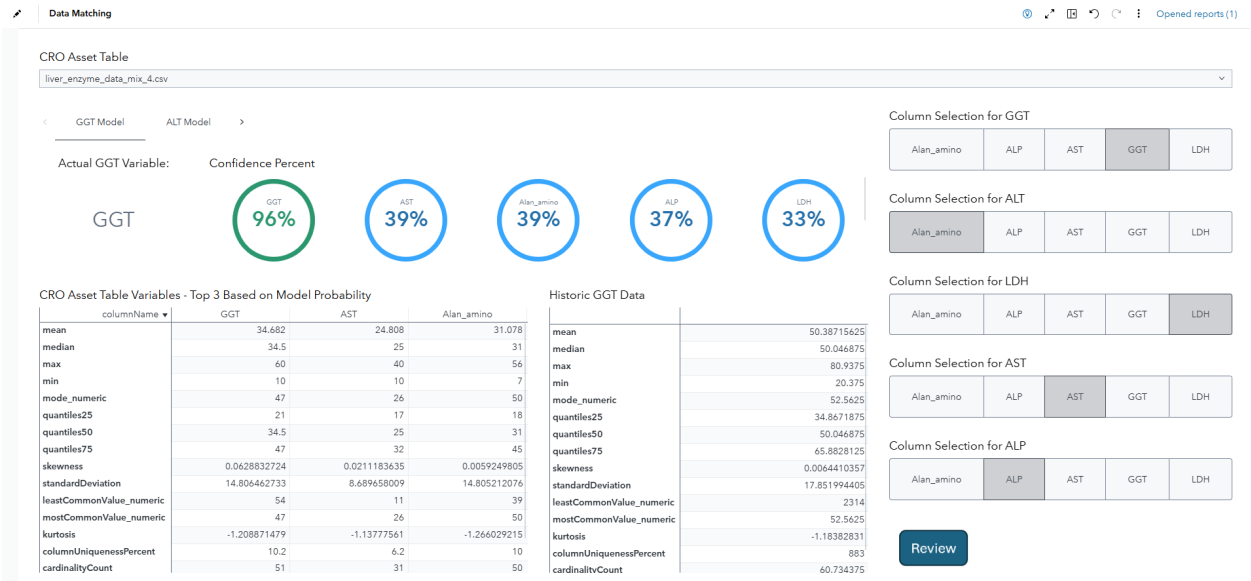
INTERACTIVE DECISION MAKING

Now that we have created an aggregate profile dataset and the score for variables in the standard, the final step is an interactive way to create quick and easy decisions. These datasets are combined into one Visual Analytics (VA) dashboard as shown in Display 8. Here the user can select which dataset to focus on by using a drop down menu. The dashboard then updates to reflect this specific dataset.

Each tab in the lower section represents details collected (the models and profiled data) for a variable in the standard. The infographic circles show the probability of each variable in the aggregate data set being the variable in the standard (based on the tab the user is in). These infographic circle change to green when the probability passes a preset threshold for easy identification. The table below the infographic circles shows the aggregate data we pulled from information catalog for the top three most likely variable matches to the standard selected by the tab. Next to that information is the overall historic profiled information for the standard variable. Being able to compare across this table provides quick reference material.

For example, in Display 8, we can compare that the variable named GGT in liver_enzyme_data_mix_4.csv, has a 96% likelihood of being variable GGT from our standard. This summary statistics also support that this might be the right match but that this data might also be on the lower end of the typic ranges. Finally, for demonstration purposes, there is a section labeled, ‘Actual GGT Variable’, this proves that GGT is actually GGT. This section would not be in the final report.

Finally, the button bars on the right provide a tracking method as the dataset variables are matched to a standard. The buttons on the button bars represent each available variable in the dataset being mapped, and each row of buttons correspond to a variable in the standard. These inputs are then pass into a web app to allow for the generation of an updated dataset at the click of a button.

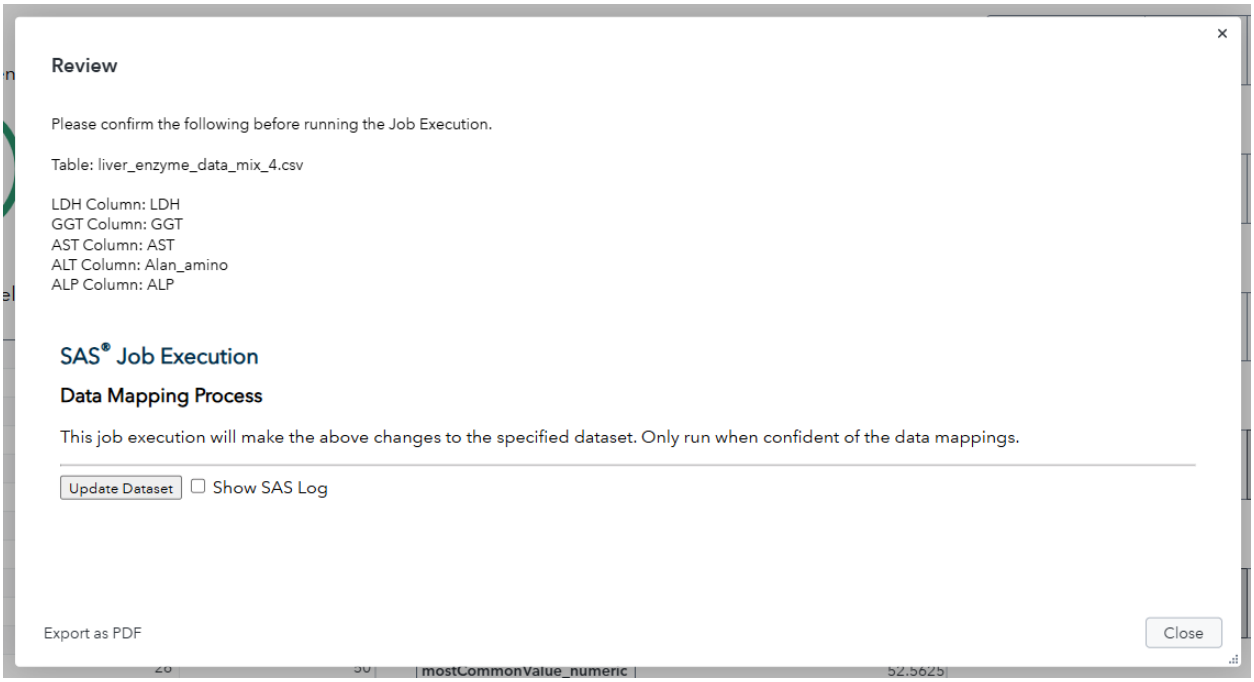


Display 8: Visual Analytics - Data Matching Dashboard

WEB APP

The method of creating the web app is two fold, using the Web Content Object in VA backed back a Job Definition built in SAS Studio. The ‘Review’ button in Display 8 opens up the pop up as shown in Display 9, this is where we have hosted the web app to provide a final review of the changes before committing them. The web app object is tied to the section labeled SAS Job Execution.

The process of creating the web app is twofold: it utilizes the Web Content Object in Visual Analytics (VA), which is backed by a Job Definition created in SAS Studio. In Display 8, the Review button opens a pop-up window, as shown in Display 9. This pop-up hosts the web app, providing users with a final review of the changes before committing them. The web app object is linked to the section labeled SAS Job Execution, ensuring seamless integration.



Display 9: Visual Analytics - Web App

The code used to create the update is provided in Program 2. It pulls in the values selected from the dashboard and assigns them as macro variables, which then guide the necessary changes. This process is highly customizable.

```
cas JE_Liver sessopts=(caslib = "CHEML");
caslib _ALL_ ASSIGN;

%JESBEGIN;

/*Load Table Param into memory*/

%let update_name = %sysfunc(cats(%scan(&Table_Param., 1, '.'),_update));
%put &update_name.;

proc casutil;
      load file="/nfsshare/sashls2/data/ChemData/Liver/&Table_Param."
      importoptions=
             (getnames="true")
```

```
            casout="under_review"
      replace;

      altertable casdata="under_review"
            rename = "&update_name."
            columns = {
                  {name= "&GGT_Param." rename="GGT"}
                  {name= "&ALP_Param." rename="ALP"}
                  {name= "&ALT_Param." rename="ALT"}
                  {name= "&AST_Param." rename="AST"}
                  {name= "&LDH_Param." rename="LDH"}
            };

      promote casdata="&update_name.";

quit;
```

Program 2: Web App Code

CONCLUSION

This project demonstrates a transformative, low-code approach to automating the quality review and harmonization of clinical trial data. By integrating out-of-the-box tools available in SAS Viya, such as Information Catalog, Visual Text Analytics, Model Studio, and Visual Analytics, we have created an end-to-end solution that minimizes manual intervention, reduces human error, and accelerates decision-making.

The system provides a seamless workflow, from profiling raw datasets to generating predictive models that match variables to a defined standard with a high degree of confidence. The use of statistical and language models enhances accuracy, while the Visual Analytics dashboard enables intuitive, interactive exploration and validation of results. This is further extended by an embedded web application that allows users to finalize and apply decisions at the click of a button.

Ultimately, this solution directly addresses the pain points of current clinical data quality reviews—manual review fatigue, inconsistency across CROs, and delayed timelines—by offering automation, transparency, and repeatability. The result is a scalable framework that supports faster time-to-registration in drug development, aligning with industry needs for efficiency, compliance, and better patient outcomes.

REFERENCES

Mary Dolegowski's Git Hub. 2025. "SD-143". Accessed April 11, 2025. https://github.com/mary-dolegowski/143

CONTACT INFORMATION

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

Mary Dolegowski
SAS Institute
mary.dolegowski@sas.com

Scott McClain
SAS Institute
scott.mcclain@sas.com