



A User Interface To Your SAS Programs



Hailemichael M. Worku, OCS Life Sciences, the Netherlands



Agenda

- Introduction
- **Case Study:** SAS Base for a Reproducible Research
- **An Alternative Approach:** via SAS EG
- Application:
 - Front-end / Interface
 - Back-end / 'The Magic'
 - Process Flow Diagram / logic
- Conclusion

Introduction



- SAS programs are usually plain text SAS scripts.
- We ran them in a manual or automated fashion.
- Challenges for 'normal' Users:
 - Lack of interactivity with users (e.g., input parameters).
 - Difficult for users who have little or no affinity for programming.
 - Users are often interested on the final report for review or submission.

Case Study 1: SAS Base for Data Analysis



The SAS System
The UNIVARIATE Procedure
Variable: Weight

Moments			
N	19	Sum Weights	19
Mean	100.026316	Sum Observations	1900.5
Std Deviation	22.7739335	Variance	518.652047
Skewness	0.18335097	Kurtosis	0.68336484
Uncorrected SS	199435.75	Corrected SS	9335.73684
Coeff Variation	22.7679419	Std Error Mean	5.22469867

Basic Statistical Measures			
Location		Variability	
Mean	100.0263	Std Deviation	22.77393
Median	99.5000	Variance	518.65205
Mode	84.0000	Range	99.50000
		Interquartile Range	28.50000

Note: The mode displayed is the smallest of 4 modes with a count of 2.

Tests for Location: Mu0=0

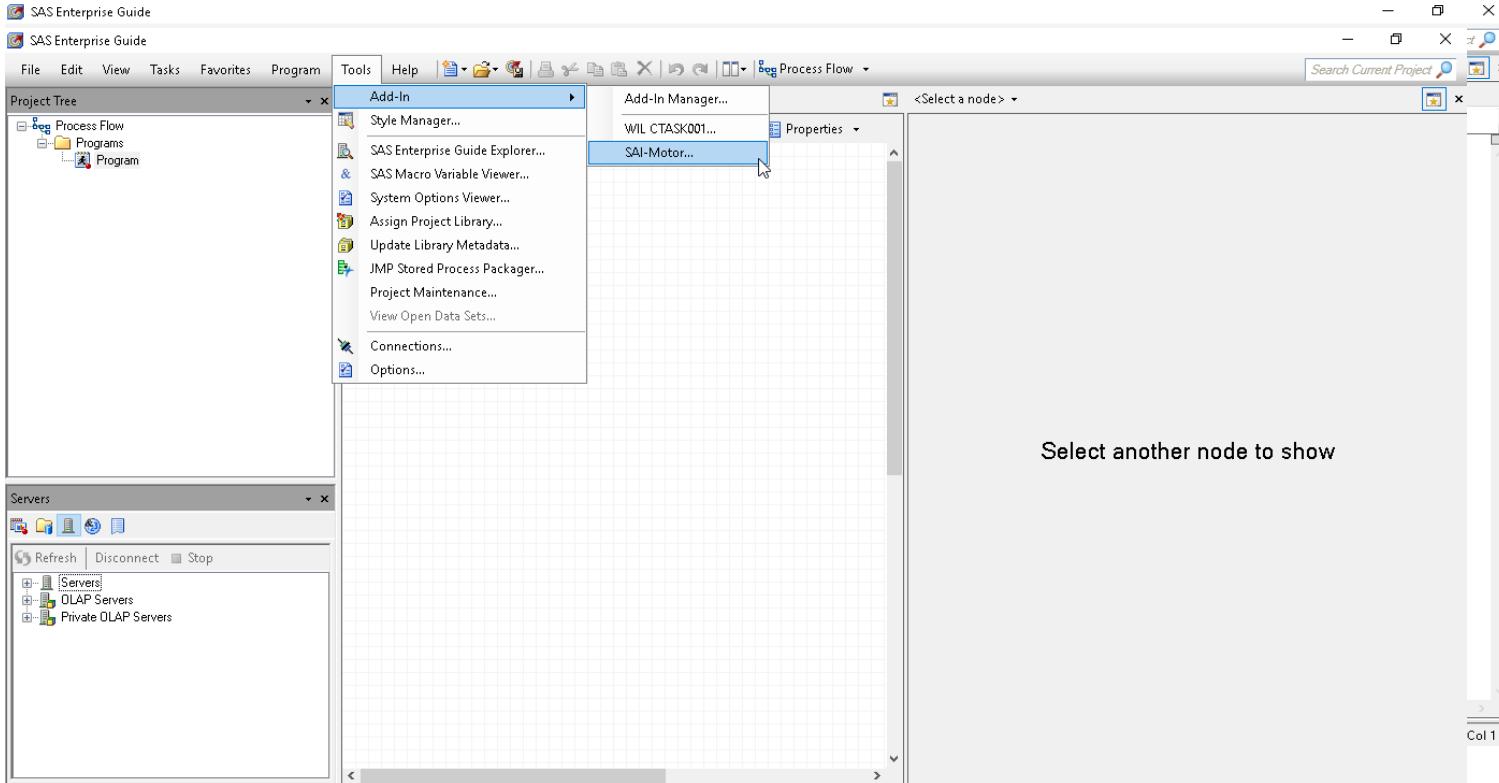


Case Study 1 (Cont'd): The Challenges?

- What If – a question by ‘normal’ users?
 - A report for weight of subjects by gender (both or single).
 - Exclude a subject whose weight value is below/above certain threshold value.
 - Import any type of class dataset and perform similar tasks.
- Do users ‘care’ about what is happening in the back-end / The ‘Magic’???



Alternative Approach: via SAS EG



Case Study 2: Temp Converter App



The Interface:

Temp Converter App (Pro)

Enter Temperature:
37.75

From / To:
°C / °F [v] Convert

Result:
99.95

Reset

The Back-end / 'Magic':

```
%let txtFld_tempInput = 37.75;
```

```
%let comboBox_fromTo = 2;
```

```
%include "temp_converter_pro.sas";
```



Case Study 3: Real-world Applications

The screenshot shows a web browser window with the URL <https://www.kaggle.com/c/titanic>. The page header includes the Kaggle logo, a search bar, and navigation links for Competitions, Datasets, Kernels, Discussion, Jobs, and a Sign In button. The main content area features a large banner for the 'Titanic: Machine Learning from Disaster' competition, with the text 'Getting Started Prediction Competition' and 'Start here! Predict survival on the Titanic and get familiar with ML basics'. Below the banner are tabs for Overview, Data, Kernels, Discussion, Leaderboard, and Rules. The 'Overview' section is active, showing a 'Description' tab with the heading 'Start here if...' and the text 'You're new to data science and machine learning, or looking for a simple intro to the Kaggle prediction competitions.' Below this is a 'Competition Description' section with the text 'The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her'.



Titanic: Machine Learning from Disaster

- Titanic Dataset:
 - Kaggle provides two set of data sets: the Training set (train.csv) and the Test set (test.csv).
 - The dataset is about 2,224 travelers and crew who were on the ship (e.g., age, gender, #siblings, #parents, fare, etc)
 - The main difference between the two input files is the response variable (survived: 1=Yes; 0=No).
- Prizes for winners and job opportunity.



The Titanic Explorer App: Front-end / Interface

Titanic Explorer App [Minimize] [Maximize] [Close]

Import Data [Load Data]

Machine Learning (ML) Train - Titanic Test - Titanic

Train - Titanic

Preprocessing

Select covariate: Age [v]

Drop missing values:

Exploratory Analysis

Summary statistics:

Graphs: Histogram [v] [Display]

Fit ML Model

Model type: Logistic regression [v]

Apply Transformation: Log [v] [Display]

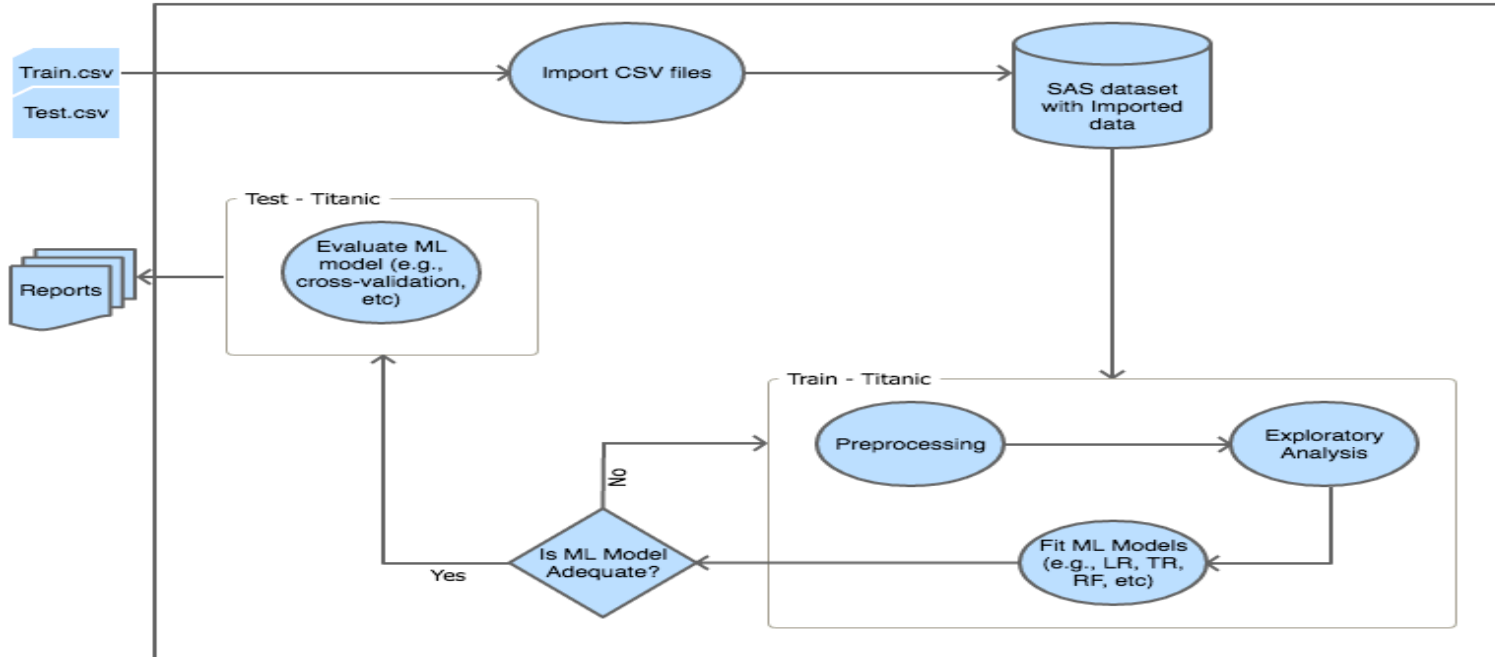
[Reset] [View Log] [Generate Report]

Output Window [v]

[Progress Bar]



The Titanic Explorer App: Process Flow Diagram





The Titanic Explorer App: Back-end / The 'Magic'

```
73
74 /*****
75 Step 0. Load Titanic raw dataset.
76 *****/
77 %load_data(raw_dsin = &intrfcToSAS_LoadArea_btnLoad.);
78
79
80 /*****
81 Titanic Explorer App.
82 *****/
83 %macro TitExpApp(MLArea = );
84
85 /* Training phase. */
86 %if %sysfunc(strip(&MLArea.)) = 1 %then %do;
87 /*****
88 Step 1. Preprocess raw Titanic data.
89 *****/
90 %preprocess_data(dsin          = work.train,
91                 varin         = &intrfcToSAS_preprArea_comboBox.,
92                 drop_missing = &intrfcToSAS_preprArea_chkBox.,
93                 debugme      = n);
94
95 /*****
96 Step 2. Explore processed dataset.
97 *****/
98 %explore_data(show_descStat = &intrfcToSAS_explArea_chkBox.,
99              graph_type    = &intrfcToSAS_explArea_comboBox.,
100              debugme      = n);
101
102 /*****
103 Step 3. Fit ML model on processed data.
104 *****/
105 %fit_MLModel(model_type      = &intrfcToSAS_fitMLArea_comboBox.,
106             apply_transformation = &intrfcToSAS_fitMLArea_comboBox.,
107             debugme          = n);
108 %end;
109 %else %if %sysfunc(strip(&MLArea.)) = 2 %then %do;
110 /* %test_MLModel(...); */
111 %end;
112
113 %mend TitExpApp;
114
115 /* Run the app. */
116 %TitExpApp(MLArea = &intrfcToSAS_MLArea_RadioBtn.);
117
118 /*****
119 Step 4. Generate report.
120 *****/
121 /*%generate_report(...); */
122
123 >>>
```

Conclusion



- Users are mainly interested on the following 4 tasks:
 - Loading study/research data.
 - Preprocess imported data.
 - Perform main tasks (e.g., explore, analyze, etc).
 - Generate a report for review/submission.
- Users are not often interested about the 'Magic'.
- Building such application/add-in could improve User's experience.
- SAS programs used for the back-end is platform independent.



Thank You!

Questions???