Paper SAS4589-2020

The Secret to Successful Current Expected Credit Loss Model Implementation

Monica Wang, Jackie Yanchuck, SAS Institute Inc.

ABSTRACT

The deadline is coming, the deadline is coming! Are your Current Expected Credit Loss (CECL) models ready to go? Financial institutions are required to have CECL models implemented as soon as 2020 or as late as 2023. Whether your model development process just started or your models have been finalized for a while now, the implementation process is still a large project to take on. This process can be just as long and complex as the model development process. So what's a financial institution to do? Fortunately, this paper is written by two model implementation experts. They have seen a wide array of models implemented for a wide range of financial institutions, and they're here to give you the tips and tricks for the most successful model implementation possible, using SAS® Model Implementation Platform (MIP) component of the SAS Expected Credit Loss (ECL) Solution.

INTRODUCTION

Often, when developing a CECL process, it is common to spend most of the time focused on the development of the model, with the implementation treated as an afterthought. Developers might not consider how long it will take to implement the CECL methodology when developing the model. For example, large amounts of historical data are often used in model development, and intense data cleaning must be performed. Both of these tasks can be very manual in nature. Do we really want these manual processes to be a part of the regular production cycle? Additionally, proper automation of data preparation, model execution, and results reporting must be set up and maintained. Have all of these been given proper consideration? In this paper, we will discuss best practices for the implementation process in the context of the SAS Expected Credit Loss solution, primarily focused on the SAS Model Implementation Platform component. This solution can expedite the implementation process while simultaneously reducing ongoing maintenance efforts and increasing transparency and auditability.

CHALLENGES

Before addressing best practices for model implementation, we want to discuss common challenges that are encountered in the model implementation process. Some of the biggest obstacles are issues in the data. These include imputation of missing values and checking data quality, because most source data requires some effort to be implementation ready. Additionally, data joining is a challenging task in the model implementation process for CECL. Vertical joins (or unions) must be configured in order to execute models and aggregate results on the financial institution's entire portfolio. For example, the loan information for both Auto and Mortgage products must be combined into one file for CECL reporting and disclosure purposes. Horizontal joins are also needed, both to bring together portfolio, economic, and model data and to combine loan attributes from different source systems.

After data challenges are addressed, large amounts of code must be written. This includes both forecasting logic that brings together multiple individual models to produce an expected loss estimate and model sensitivity testing and attribution analysis code. For each methodology that is developed, this process must be repeated. Often, the resulting code is complicated or repetitive. Unless the code is optimized (which can be a lengthy process), the model execution process may be very time and resource intensive, because it requires a large number of calculations across the financial institution's entire portfolio.

The challenges don't end after the CECL implementation process is completed. Auditing must occur on the entire process, either by internal model validation teams, external auditors, or both. This task can be cumbersome and time-consuming if the code is convoluted and difficult to understand. Additionally, convoluted code is difficult for the end user to utilize, and for others update the existing process.

DESIGN

Many of the challenges discussed in the previous section can (and should) be addressed in the design phase of both model development and model implementation. Different challenges can be addressed in each of these phases. In model development design, be sure to consider ramifications on model implementation design. For example, data preparation, model creation, and model execution code should be separated out, because each one serves a different purpose. Only data preparation and model execution need to be incorporated into the production cycle. If all three pieces of code are intertwined, part of the model implementation process is pulling out the necessary pieces of code. This effort is unnecessary if the code is appropriately segmented in the first place. Another consideration is the availability of variables when forecasting. For example, the occurrence of a natural disaster may have a high correlation with probability of default. However, it is not an appropriate variable to incorporate into a probability of default model because forward-looking data is not accessible for this predictor.

Additionally, you should think about how the processes will be automated as part of the production cycle. For example, if some data preparation was performed manually for model development, you need to consider how will it be performed as part of the production process. If it will be difficult to automate, you could consider redesigning or removing this component of data preparation.

In the case of model implementation, consider differences between the input data for model development and for the production cycle execution – are they different? If so, how will that data be brought into the model execution environment, and will any processing, such as joins, need to be performed?

SAS Model Implementation Platform has many benefits for the simplification of model implementation design, and it reduces the amount of code to be written for each methodology. The only code that needs to be written is the extraction and preparation of the data from the source system and the actual model implementation code. No code needs to be written to perform joins between input data sets, to combine outputs from the execution of multiple methodologies, to run attribution analysis, or to perform sensitivity analysis.

While the code to be written in SAS Model Implementation Platform is limited, there are still important design questions to consider in order to best utilize the solution. Below are some examples that affect how the methodology is implemented:

- Is this an aggregate level model or loan level model?
- Are models segmented in any way? What is that segmentation logic?
- Does this model involve transition matrices, and if so, what kind?
- Is there data to be sourced outside of the portfolio or the economic data?
- How will model variables change over time?
- Are there any complex calculations to document outside of the models?
- What are the desired output variables, and how are they calculated?

IMPLEMENTATION

In this section, we will address the answers to these questions, and show how to turn the design into an implemented methodology, using SAS Model Implementation Platform. Here are the effects of some possible answers to the questions posed:

- If an aggregate level model is used, it should be incorporated into a scoring model group within SAS Model Implementation Platform. This means that the model won't be executed for each loan, but rather for each distinct aggregate-level value (for example, for each distinct customer ID or each group of loans that share common attributes). All other loan level models are incorporated into an evaluation model group.
- If the models are segmented, but similar methodology is used among the segments, there is no need to break the segments into different model groups. Instead, the segmentation logic can be incorporated into the model group, and the appropriate models can be run.
- Dynamic and static transition matrices are implemented differently within SAS Model Implementation Platform. A static transition matrix is stored as a model object. For a dynamic transition matrix, the appropriate to and from states must be specified in each of the transition matrix's component models, and then the matrix is formed in the model group.
- For sourcing data other than portfolio or economic data into SAS Model Implementation Platform, data sets called risk data objects are used. Parameter matrices are used to load static lookup tables. Value datasets are used to load loan or aggregate level information that changes over time in a predefined manner.
- Within SAS Model Implementation Platform, model variables are defined differently based on how they change over time. This is done for efficiency purposes, so variables are only initialized as often as necessary. For example, if a variable (such as origination credit score) doesn't change over time, it is defined as static. If a variable (such as a loan's age) changes over time in a manner already known, it is defined as age-indexed. If a variable (such as months since last delinquency) changes over time, but we do not know what that value will be, or if the variable needs to be defined in the model group logic, it is defined as simulation.
- It is important to document any additional complex calculations performed outside of the models, so that the implementation can be more easily understood by an outside party (for example, the model validation team).
- The more output variables that are needed, the longer the implementation code will take to run. Therefore, it is necessary to define which output variables are needed for the final production cycle, and which variables may only be necessary for ad hoc analyses.

In working through writing the implementation logic in SAS Model Implementation Platform, there are some best practices that should be followed. These are listed below and are broken into two sections – best practices related to efficiency and all other best practices. Many of these best practices were provided by Shannon Clark, an SAS Model Implementation Platform developer.

EFFICIENCY BEST PRACTICES

- Minimize the number of string comparisons, because it is more resource-intensive to perform string comparisons than numeric comparisons. If a string comparison is needed, create a numeric indicator in the LOAN_INIT block, and use the numeric indicator in the MAIN block. Depending on the number of simulations or forecast horizons run, this reduces run-time, as well as memory usage.
- Appropriately define model variables as static, age-indexed, or simulation, as described in the previous section.
- Avoid utilizing the dynamic_array subroutine if array lengths between loans are similar, and instead set the array length to be the longest possible value.
- Minimize the use of INTNX and INTCK functions. If needed, use them in the LOAN_INIT block or define other appropriate logic.
- Terminate loans by setting the system variable _terminate_rep_ equal to 1.
- If appropriate, call entire rows or columns of parameter matrices, rather than individual cells.

OTHER BEST PRACTICES

- Avoid bleed-over between scenarios and loans by clearing arrays using the call zeroMatrix subroutine, and by resetting temporary variables as appropriate. This is especially important since SAS Model Implementation Platform retains values of all temporary variables.
- Utilize select statements instead of multiple if-then-else statements.
- Output and check return codes from model execution and parameter matrices to avoid silent errors.
- Use transition matrices to run multiple models at once, rather than making multiple individual model calls.
- Use functions for logic pieces common across model groups to ensure consistency.
- Use value datasets instead of cash flow datasets, because cash flow datasets have more stringent requirements.

CONCLUSION

As evidenced by this paper, there is a lot to consider when implementing a CECL methodology. The hardest work does not stop when model development completes, and often implementation can be as involved as development. It is important to give the model implementation the appropriate time and consideration, so that the output that is run as a part of the production cycle is efficient, transparent, and auditable as CECL goes into effect.

REFERENCES

"SAS® Expected Credit Loss." SAS Institute Inc. Available https://www.sas.com/en_us/software/expected-credit-loss.html. Accessed November 1, 2019.

ACKNOWLEDGMENTS

A special thank you to Shannon Clark for his guidance and contributions to this paper.

RECOMMENDED READING

- *SAS® Model Implementation Platform: User's Guide*
- *SAS® Risk Dimensions® and SAS® High-Performance Risk: Procedures Guide*

CONTACT INFORMATION

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

Monica Wang
SAS Institute, Inc.
919-531-4646
Monica.Wang@sas.com

Jackie Yanchuck
SAS Institute, Inc.
919-531-4346
Jackie.Yanchuck@sas.com