The first scenario brings some different possibilities of solutions at various levels of sustainability for the organization. Since SAP allows for some basic SQL functionality the output model coefficients could be placed into tables. You could then create a calculated field that could replicate the response variable to flag customer records for the load offer, and the prediction replicated within SAP. The main issue of this is if at any point the model would need to be retrained or adjusted it would require manual intervention. Solutions like the one proposed brings on a great deal of technical debt, meaning it requires a great deal of manual intervention which will take away man hours from other projects.

A more elegant solution to the problem would require the potential purchase (depending on contract etc.) of the appropriate module to integrate with SAP. One possible solution would be the analyst create two pipelines using the Machine Learning Scenario Manager in SAP. A training pipeline and a prediction/inference pipeline with REST-API. For the training pipeline the analyst would use the R Producer template and run the R script into the R client then he will be able to save and execute the pipeline from the scenario manager. The pipeline will save the model to the SAP data lake. Next the analyst would create another pipeline with R. Run the R script into the R Consumer pipeline and then save and deploy from the Machine Learning Scenario Manager. Then there should be an option to select the training data model which will allow a user to input into the prediction interface and get if an individual is a risk or not.

The scenario on question 2 offers more flexibility just by having the data warehouse as an intermediary between R and the CRM. The data could easily be piped into R and ran through the model (exclude the ID’s) then the results could be appended back to the data and pushed back into a data warehouse table. Depending on the level of integration from the CRM, we could filter the results being written back to the data warehouse using dplyr. Also, depending on the spec we could even archive who received offers after the CRM triggers the discount to then do some analytics on how effective our discount recommendation is.

Just comparing the two scenario’s we can see the importance of forward thinking when selecting systems. Many times, analytics/ML is an afterthought and the importance of interoperability is not considered. Going back to the prior discussion on technical debt this can lead to organizations having to make costly to have to modify systems after deployment to accomplish organizational goals.

Buying into any ecosystem whether it be Azure, Oracle, SAP or a host of others could present unique challenges for any organization if the decision is made to use the proprietary tools with these platforms. Fortunately, many platforms are offering integration with open source programming languages such as R and Python. Making decisions like this not only limit you on products you can integrate with down the road, but also will limit your talent pool if you are looking for very specific skills to integrate successfully in your organizations.

Whatever platform or solution is decided upon it is important to design your systems to easily be adapted and integrated for new and emerging trends. For machine learning we need to be able to connect to data in order to train and test our model. Then we need an effective way to deploy our model and productionalize our work. Finally, we need a way to communicate our results either via API or visualization/reporting tool.

<https://blogs.sap.com/2019/12/05/sap-data-intelligence-create-your-first-ml-scenario-with-r/>