Campaign Management Template implemented on SQL Server R Service

**Introduction:**

Campaign Response Modeling is an important machine learning application. This template showcases the use of Random Forest and Gradient Boosting to model for Campaign Responses. The Campaign Response which is a binary variable is predicted for each channel, day (of week) and time (of day) for each lead.

This template uses simulated data from the insurance industry to model the campaign response for an acquisition campaign. The model uses predictors like demographic details of the lead, historical campaign performance and Product details. The model predicts the probability of a lead making a purchase from each channel, at various times of the day and each product, for every lead in the database. The final recommendation for targeting users is decided based on the combination of Channel day (of week) and time (of day) with the highest probability of making a purchase.

For customers who prefer an on-premise solution, the implementation with SQL Server R Services is a great option, which takes advantage of the power of SQL Server and RevScaleR. In this template, we have implemented all steps in SQL stored procedures, which includes data cleaning, data preprocessing, feature engineering, model training, scoring and evaluation.

All these steps can be executed on SQL Server client environment (such as SQL Server Management Studio), as well as from other applications. We provide a Windows PowerShell script which invokes the SQL scripts and demonstrate the end-to-end modeling process.

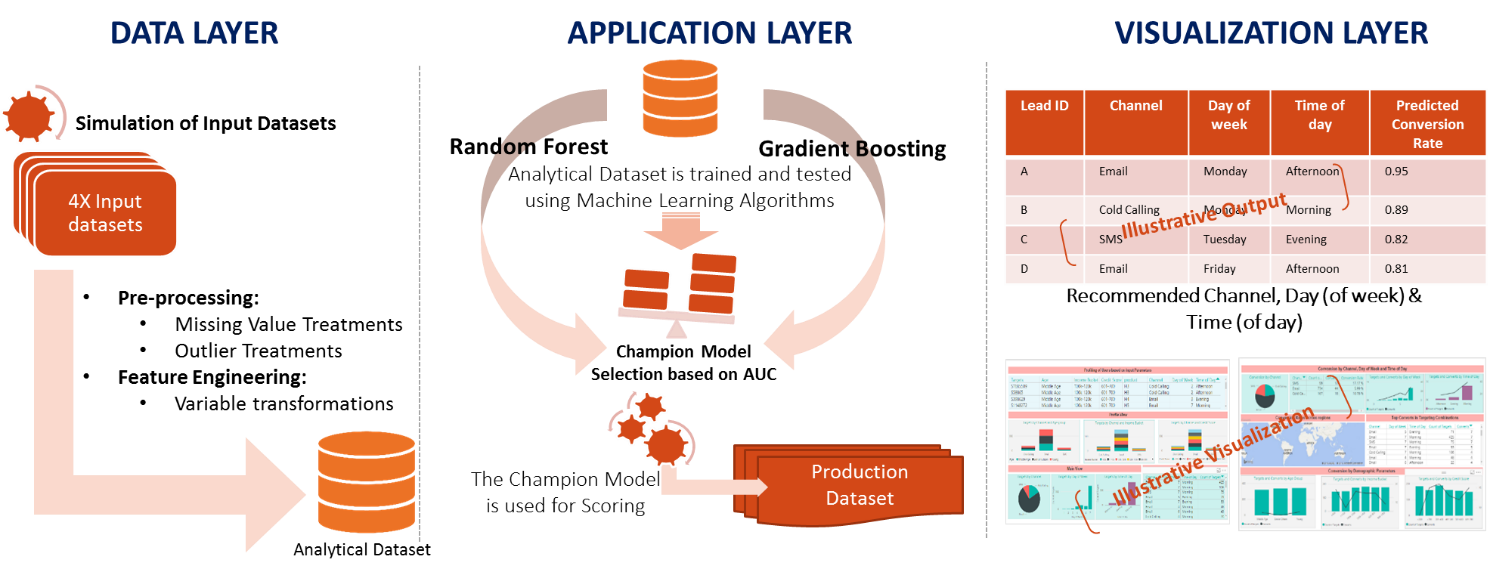
**System Requirements:**

To run the scripts, it requires the following:

* SQL server 2016 CTP 3 with Microsoft R server installed and configured;
* The SQL user name and password, and the user is configured properly to execute R scripts in-memory;
* SQL Database which the user has write permission and execute stored procedures;
* For more information about SQL server 2016 and R service, please visit: <https://msdn.microsoft.com/en-us/library/mt604847.aspx>

**Solution:**

The solution has 3 layers as show in the below figure. The Data Layer simulates the input datasets and creates the Analytical Dataset. The Application Layer trains the Machine Learning algorithms and selects the champion model. Finally, the Visualization layer scores the production datasets

****

Each of these layers are further split into smaller steps. There are 9 steps in total. The steps 0,1, 2, 3 and 4 are automated using the PowerShell script **Analytical Dataset Creation.ps1**.The steps5,6 and 7 are automated using the PowerShell script **Model Development.ps1**. The final step (step 8) which for scoring the production dataset is automated using the PowerShell script **Scoring.ps1**. All of these PowerShell scripts can be found in the SLQR folder

* Step 0: Input data Schema Preparation
* Step 1: Input data preparation
* Step 2: Preprocessing
* Step 3: Feature Engineering
* Step 4: Analytical Dataset Creation
* Step 5: Model Training
* Step 6: Model Statistics Calculation
* Step 7: Champion Model Selection & Scoring
* Step 8: Production Scoring & Visualization though PowerShell

**Step 0: Input data Schema Preparation**

In this step the schema for the input datasets is created. This step can be skipped if the data is simulated instead of imported

**Step 1: Input data preparation**

The template takes four raw files as input. Description of each of these input files can be found in the Data folder. These datasets can be created by running the SQLR scripts. The scripts will simulate the data along with outliers and missing values to mirror real life datasets. This step can be skipped and the static input datasets can be imported manually (or using the **Data Import.ps1** Powershell script) if required

|  |  |  |
| --- | --- | --- |
| SQLR Script Name | Database object Name | |
| step1(a)\_campaign\_detail.sql | | Campaign\_Detail |
| step1(b)\_product.sql | | Product |
| step1(c)\_lead\_demography.sql | | Lead\_Demography |
| step1(d)\_market\_touchdown.sql | | Market\_Touchdown |

**Step 2: Preprocessing**

Preprocessing is performed on two of the input files. To showcase preprocessing, the outliers and missing values simulated in the previous step are treated here. The script to perform these operations can be found in the SQLR folder

1. **step2(a)\_preprocessing\_market\_touchdown.sql**

This script treats the Comm\_Latency variable for outliers. The lower extremes are replaced with the difference of Mean and Standard Deviation. The higher extremes are replaced with the sum of Mean and two Standard Deviations

**Input database object: Market\_Touchdown**

**Output database object: Market\_Touchdown**

1. **step2(b)\_preprocessing\_lead\_demography.sql**

This script treats the No\_Of\_Children, No\_Of\_Dependants, Highest\_Education and Household\_Size variable for missing values. All missing values are replaced with the mode of the respective attributes

**Input database object: Lead\_Demography**

**Output database object: Lead\_Demography**

**Step 3: Feature Engineering**

Feature Engineering is performed on market touchdown dataset. The table is aggregated at a lead level, so variables like channel which will have more than one value for each user are pivoted and aggregated to from variables like SMS count, Email count, Call Count, Last Communication Channel, Second Last Communication Channel etc. The script to perform these operations can be found in the SQLR folder with the file name **step3\_feature\_engineering\_market\_touchdown.sql**

**Input database object: Market\_Touchdown**

**Output database object: Market\_Touchdown\_Agg**

**Step 4: Analytical Dataset Creation**

In this step the Analytical dataset which will eventually be used for modelling is created. Further, the analytical dataset is split into a train and test dataset with a 70-30 split. The script for this step can be found in the SQLR folder with the file name **step4\_ad\_creation.sql**

**Input database object: Market\_Touchdown\_Agg, Lead\_Demography, Campaign\_Detail, Product**

**Output database object: CM\_AD, CM\_AD\_train, CM\_AD\_test**

**CM\_AD**

|  |  |  |
| --- | --- | --- |
| Data Fields | Description | Creation Logic |
| Lead\_Id | Unique identifier of lead |  |
| Age | Age group of the lead |  |
| Marital\_Status | Marital status of the lead |  |
| Credit\_Score | Credit Score Range of the lead |  |
| Annual\_Income | Annual Income Range of the lead |  |
| No\_Of\_Dependents | Number of dependents the lead has |  |
| Highest\_Education | Highest Education of the lead |  |
| Source | Source from which the user came into the database |  |
| Product | Product Name |  |
| Category | Product Category |  |
| Term | Number of months of coverage |  |
| No\_Of\_People\_Covered | Number of people covered in the policy |  |
| Premium | Premium to be paid by the user |  |
| Payment\_Frequency | Payment frequency of the product |  |
| Amt\_On\_Maturity\_Bin | Bucketed Dollar Amount on Maturity |  |
| Campaign\_Name | Campaign Name |  |
| Sub\_Category | Sub Category of the Campaign |  |
| Campaign\_Drivers | Drivers for the Campaign |  |
| Call\_For\_Action | Objective of the campaign |  |
| Tenure\_Of\_Campaign | Tenure of the campaign |  |
| Action\_Day | Integer values showing the day of the week the lead was contacted | Derived from Day of Week column in Market Touchdown |
| Action\_Time | Time of day when the lead was contacted | Derived from time of Day column in Market Touchdown |
| Last\_Channel | Channel via which the user was contacted last | Built by ranking the columns Channel 1, Channel 2 and Channel 3 Communication ID |
| Second\_Last\_Channel | Channel via which the user was contacted the last but one time | Built by ranking the columns Channel 1, Channel 2 and Channel 3 Communication ID |
| Email\_Count | Number of emails the user has received in the past | Built by counting the numbers of emails the lead has received in the previous year |
| Call\_Count | Number of calls the user has received in the past | Built by counting the numbers of calls the lead has received in the previous year |
| SMS\_Count | Number of SMS the user has received in the past | Built by counting the numbers of SMS the lead has received in the previous year |
| Comm\_Frequency | Total number of times the user was touched | Built by counting the numbers of times the lead has received marketing in the previous year |
| Conversion\_Flag | Final dependent variable with the value ‘1’ indicating a successful purchase |  |

Analytical Dataset

Feature Engg.

Data Cleaning

Market Touchdown

lead Demography

Product

Campaign

**Step 1: Input data preparation**

**Step 5: Model Training**

In this step, two models are built using 2 statistical techniques on the training Dataset. The scripts to build the model using each of the techniques can be found in the SQLR folder.

1. **Step5(a)\_model\_train\_rf.sql**

This file contains the script to build Random Forest on the training dataset. The parameter ‘mTry’ which signifies the number of variables sampled at each split is set as 5 (default value is square root of the number of variables in the dataset). The parameter ‘nTree’ which signifies the numbers the number of trees to grow is set as 500. The complexity parameter ‘cp’ is used to control the size of the decision tree and to select the optimal tree size. If the cost of adding another variable to the decision tree from the current node is above the value of cp, then tree building does not continue. cp is set as 0.00001

**Input database object: CM\_AD\_train**

**Output database object: model\_rf**

1. **Step5(b)\_model\_train\_gbm.sql**

This file contains the script to build the Gradient Boosting model on the Analytical dataset. The parameters are set as follows

|  |  |
| --- | --- |
| Parameter | Value |
| Learning\_rate | 0.2 |
| minSplit | 10 |
| minBucket | 10 |
| nTree | 500 |
| Seed | 5 |
| LOSS Function | Multinomial |
| computeContext | RxLocalParallel |

**Input database object: CM\_AD\_train**

**Output database object: model\_gbm**

**Step 6: Model Statistics Calculation**

Once the models are trained in the previous step, the test dataset is scored on both the model algorithms and the AUC and Accuracy for each model is calculated. The script for this step can be found in the SQLR folder with the file name **step6\_models\_comparision.sql**

**Input database object: CM\_AD\_test, model\_rf, model\_gbm**

**Output database object: model\_statistics**

**Step 7: Champion Model Selection & Scoring**

In this step the champion model is selected based on the metrics calculated in the previous step. Once the champion model is selected, the entire analytical dataset is scored using the champion model and the final scored data is created which will be used as the input for the PowerBI dashboard. The script for this step can be found in the SQLR folder with the file name

1. **Step7(a)\_scoring\_leads.sql**

In this step the champion model is identified and the analytical dataset is scored on the champion model

**Input database object: CM\_AD, model\_statistics**

**Output database object: lead\_list**

1. **Step7(b)\_lead\_scored\_dataset.sql**

In this step the final scored dataset is created. This is will be used as the input for the PowerBI dashboard

**Input database object: lead\_list**

**Output database object: Lead\_Scored\_Dataset**

Random

Forest

Gradient Boosting

Analytical Dataset

Champion Model

Model statistics comparison

Random Forest model statistics

Gradient Boosting model statistics

**Step 8: Production Scoring & Visualization though PowerShell**

The PowerShell script will trigger the champion model to run on the production data. The script can be found in the SQLR folder with the file name **Scoring.ps1**