White Paper
Creating Complex Metrics in Genesys Predictive Routing

There are two ways to work with multiple metrics in Genesys Predictive Routing (GPR). One is by use of *composite Predictors*, which are configured using the GPR web application. The other is to perform some arithmetical calculation on metric values during the data preparation phase, outside the GPR application, and import a Dataset that includes the resulting *complex metric*.

- Composite Predictors are explained in the Help file: About Composite Predictors.
- Complex metrics are discussed in this White Paper.

Composite Predictors offer the following advantages over complex metrics:

- **Efficient configuration:** It is easier to create the desired calculation when you combine separate, already-created Predictors.
- **Maximized data usage:** A key advantage of starting with separate Predictors, each of which can draw on a different Dataset, over pre-computing the final metric in a common Dataset, is that the resulting composite Predictor is based on a broader range of data than a single Dataset.

Important

The Lift Estimation report, which evaluates lift based on *Models*, does not support composite Predictors. If you need to assess potential lift based on the output from multiple metrics, create a Model from a complex metric.

Complex metrics are based on multiple outcomes. After performing the calculation that you have determined will produce an output value correctly representing the relationships among the selected metrics, add this final output value as a separate column to your Dataset before uploading the Dataset to GPR. GPR can then handle this metric like any single-KPI metric.

Creating an Effective Arithmetical Expression

To create a complex metric, select two or more metrics, determine how they are related, then create an appropriate arithmetical expression to resolve them into a single value. The arithmetical expression should depend on the relative importance of the metrics you select. It should also account for factors such as the following:

- Differing amounts of data for the various metrics.
- Different types of values (integer, float, Boolean).
- The features with the strongest impact on the outcome of each metric might have little or no overlap. This could confuse the effort to identify the critical features to focus on for improvement.

- The difficulty of creating a straightforward arithmetical expression to find equivalence between the simple Predictors.

  For example, you might want to increase first-contact resolution (FCR), reduce average handle time (AHT), and reduce the rate of escalations. You need to know how they are related, since the value of one metric might affect the value of another. For example, you might find that FCR improves as AHT rises, so that a better AHT result damages your efforts to improve FCR.

Creating a sound arithmetical expression depends your own expert judgment and understanding of your business requirements and your environment. It also depends on high-quality data; that is, data that is complete, contains a balanced representation of agent performance, and is relevant to the metrics you are targeting.

Warning

Large coefficients (weights), certain transformations (like logs and exponents), and other manipulations with metrics can significantly change routing.

Examples

The following examples provide some guidelines for how to approach the problem of constructing a well-thought-out complex metric.

Example 1: Finding the optimal discount to retain a customer

In this example, we combine FCR and AHT to balance quality and speed. The general idea is to reduce both metrics to the same units and use the result as a single metric. The formula to achieve the new metric is:

```
( - 1 ) * ( AHT - 300 * ( FCR - 1 ) )
```

where AHT is in seconds and FCR equals 1 if the call was resolved immediately and 0 if additional calls followed. Using this expression, an unresolved interaction leads to a score penalty of 300 seconds. Multiplying by –1 converts the result to one where a lower number represents a better outcome.

Example 2: Finding the optimal discount to retain a customer

In this example, we pick an optimal discount to offer a customer in a retention process after the customer's initial cancellation request.

If the customer stays, the final metric, *y*, is calculated using the following formula:

y = *base_customer_value (bcv)* − discount

If the customer leaves, y = 0.

GPR builds a Model to predict the expected final outcome for every discount amount and then recommends the discount associated with the highest expected final value.

Example 3: combining metrics with opposite target objectives

Assume that the target metrics for the respective Predictors are the following:

- Predictor1 - NPS
- Predictor2 - CSAT
- Predictor3 - TALK_DURATION

All three produce numeric results. You might write an expression such as the following:

(Predictor1 + Predictor2) / Predictor3

In this example, the final agent score is:

(predicted NPS + predicted CSAT) / predicted TALK_DURATION

That means that agents with higher NPS and CSAT numbers end up with higher final scores if their TALK_DURATION is lower.

Example 4: Modifying weights to reflect your business expert judgment

In this example, we use the same simple Predictors as in example 1, but adjust the expression to reflect your informed judgment about the relative values of the different metrics. You might create the following expression, which weights NPS to be twice as important as CSAT:

(2 * Predictor1 + Predictor2) / Predictor3

Example 5: Categorization using a threshold

You might use different simple Predictors depending on a context variable passed in a scoring request. You can create a composite Predictor that would use one simple Predictor for customers in a high-value category (value >= 50) and a different simple Predictor for lower-value customers (value <= 50).

The idea is to offload this logic from the strategy into a Predictor, which reduces the need to modify your strategy. To create such a Predictor, define the following expression for your composite Predictor:

int(value > 50) * Predictor1 + int(value <= 50) * Predictor2

The logic here is the following. The expressions (value > 50) and (value <= 50) have Boolean results, true< or false. Using the function int(value > 50) and int(value <= 50) , we convert the text-format Boolean values, true/false, to the equivalent in digits, 1/0. Then, whichever Predictor value is multiplied by 0 is ignored and the remaining Predictor is used for routing.

So if the value of the metric is 61, the first expression is true/1, and the second is false/0. Predictor 1's value is multiplied by 1 and produces a result. Predictor 2's value is multiplied by 0, which is not a valid arithmetical operation, and the value is ignored.