# Validación de un modelo: Respuesta a una campaña de mail usando Naïve Bayes

La calidad de un modelo de Machine Learning no se evalúa exclusivamente con los resultados que este genera durante su entrenamiento y validación. La aplicabilidad de cualquier modelo debe considerar múltiples métricas que demuestren su efectividad en la realidad.

Evaluar incorrectamente un modelo podría generar expectativas irrealistas de su funcionamiento, las cuales harán aparecer problemas en la fase de producción, al utilizar el modelo con datos de la realidad.

Por tanto, el desempeño real de un modelo podría no ser aceptable y en muchos casos generar daños elevados para los clientes, incluyendo dinero y vidas.

Un error frecuente de los analistas, es sobredimensionar el funcionamiento de un modelo en base al porcentaje de precisión que arroje. El objetivo de este documento será evaluar distintas técnicas de validación de modelos de Machine Learning para un caso particular.

## Escenario

Para este caso, utilizaremos la herramienta de Rapidminer para generar datos de ejemplo. El objetivo es predecir si una persona responderá a una campaña de marketing por mail utilizando atributos demográficos.

Para ello, existe un operador en Rapidminer “Generate Direct Mailing Data”. Este operador utiliza dos parámetros:

* Número de ejemplos a generar. En este caso usaré 10000.
* Semilla local. Activar este parámetro me permitirá repetir el ejercicio manteniendo los datos generados.

## Preparación de Datos

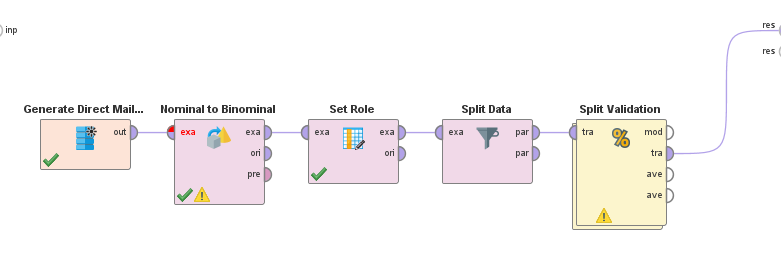
El operador de generación arroja un dataset del tamaño indicado con los siguientes predictores:

* Nombre
* Edad
* Estilo de vida (Tres categorías: Activo, Cómodo o Saludable)
* Código Postal
* Estado Civil
* Tipo de vehículo (Dos categorías: Práctico o Costoso)
* Deporte (Tres deportes distintos: Bádminton, Fútbol y Atletismo)
* Salario
* Respuesta al mail

Antes de comenzar a trabajar, fue necesario realizar algunos ajustes. En primer lugar, la variable a predecir (Si la persona realmente responderá a la campaña por mail) tiene dos posibles valores Response / No Responde, por lo que se puede tratar esta variable como Binomial.

El dataset completo está etiquetado, por lo que podremos dividirlo en dos para separar en Entrenamiento y Test. Aplicaremos la regla del 80/20 y además configuraremos un Split Validation para el dataset de entrenamiento.

En este caso, la validación tendrá una proporción 70/30. El flujo resultante es el siguiente:



## Modelo

Al enfrentarnos a un problema supervisado de clasificación binaria, disponemos de una gran variedad de modelos para probar. Sin embargo, como el objetivo es analizar las métricas de Performance, optaremos por un modelo sencillo de Naive Bayes sin pensar mucho en cuál será el más adecuado.

Dentro del operador de Split Validation, ubiqué el operador de Naive Bayes, aplicación del modelo y luego un operador de Performance activando las siguientes opciones:

* Accuracy
* False positive
* False negative
* True positive
* True negative
* Sensivity
* Specificity
* AUC

Por último, utilicé el dataset reservado como Hold-out para testear el modelo generado. A nivel principal del flujo, se debe agregar un operador de Apply Model con el Dataset de test.

Como última medida de performance, agregué una Lift Chart a la salida del Apply Model. Este gráfico apuntará al valor de la clase a predecir “Response”.

## Resultados

El proceso de Rapidminer genera la siguiente matriz de confusión para los resultados del dataset de entrenamiento:

|  |  |  |
| --- | --- | --- |
|  | Response reales | No Response reales |
| Response Predichos | 56 | 33 |
| No Response Predichos | 15 | 136 |

Si tomamos como clase positiva a los valores “Response”, obtendremos los siguientes datos:

* Falsos positivos (FP): 33
* Falsos negativos (FN): 15
* Verdaderos positivos (TP): 56
* Verdaderos negativos (TN): 136

Para estos valores, podemos verificar las métricas teóricas y contrastarlas con los resultados de Rapidminer.

Sensibilidad: *TP / (TP + FN)*

56 / (56 + 15) = 56 / 71 = 0.788 => **78.8%**

Especificidad: *TN / (TN + FP)*

136 / (136 + 33) = 136 / 169 = 0.804 = **80.4%**

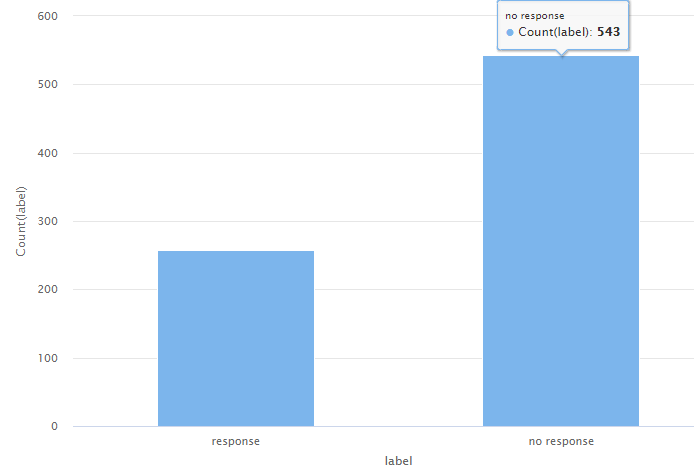
Estos valores son idénticos a los calculados automáticamente por Rapidminer:



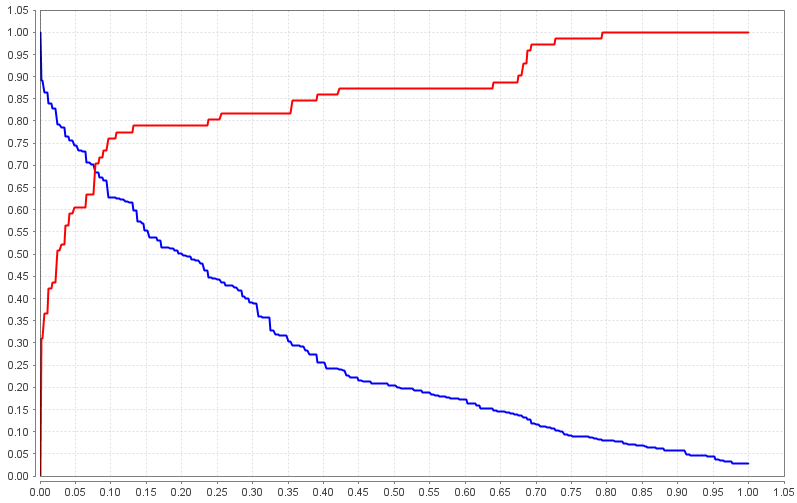
Resulta interesante analizar los valores de Class Recall. Puntualmente, estos valores representan la proporción en que una clase dada de los datos fue predicha correctamente.

Este valor tiende a ser influenciado por la representación de la clase en el Dataset. Es decir, el modelo podrá comportarse adecuadamente en general, pero muy deficientemente para una clase en particular. Este efecto se maximiza cuando otras clases están sobrerepresentadas en el Dataset. Cuando esto sucede, se dice que el Dataset está desbalanceado.

En este caso, el Recall para los “No Response” es mayor que el Recall para los “Response”. Si analizamos la representación de cada clase en el dataset inicial, podemos ver que realmente existe una relación entre los valores resultantes y la proporción de las clases.



Con respecto a la curva ROC y el valor AUC, podemos decir que el modelo se comporta relativamente bien. El valor resultante del área bajo la curva es de 0.861. Supera el valor de la elección aleatoria, donde el área sería de 0.5.



Tomando en consideración este mismo concepto de mejora sobre la selección aleatoria, la confianza de predicción del modelo va a ser distinta para cada uno de los valores que clasifica.

A partir de esta información, centrándonos en los verdaderos positivos, es posible construir un gráfico que muestre como varía la mejora del modelo en función de la confianza.

De esta manera, es posible saber la mejora sobre la elección aleatoria, en definitiva, que tan bien funciona un modelo, tomando parte de la población total.

En este caso encontré que la cobertura alcanza el 70% en la tercera barra del gráfico. Este dato es interesante porque significaría que un 30% de la población es necesaria para lograr un 70%.

