AIEP
de la Universidad Andrés Bello

Taller de analítica en los negocios

Analítica en marketing

Escuela de Administración y Gestión Empresarial

Directora: Lorena Baus Piva

Elaboración

Experto disciplinar: Tomás von Bischoffshausen Gariazzo

Diseñador instruccional: Óscar González Cantin

Editor instruccional: David Villagrán Ruz

Validación

Experto disciplinar: Andrés Montecinos Rüth

Jefa de diseño instruccional: Sandra Betancur Cordero

Equipo de desarrollo

AIEP

AÑO

2025

Tabla de contenidos

Aprendizaje esperado de la semana

Aplican modelo de predicción de *churn* con algoritmos clasificadores a través de herramientas Python, considerando análisis de datos en el contexto de estrategias de marketing.

Fuente: Envato Elements (s. f.)

Introducción

¿Alguna vez te has preguntado cómo los modelos de análisis de datos determinan qué factores son más relevantes para predecir comportamientos o resultados?

Esta semana pondrás en práctica tus conocimientos en la creación de modelos clasificadores al servicio de la predicción de *churn* (abandono de clientes).

El análisis de *churn* es una técnica muy relevante en la gestión empresarial, especialmente en sectores como telecomunicaciones, banca y suscripciones digitales, pues una reducción efectiva en la tasa de *churn* puede mejorar significativamente la rentabilidad, dado el alto costo de adquisición de nuevos clientes.

El término *churn* se refiere a la pérdida de clientes o usuarios en un período determinado. La literatura ha destacado el uso de modelos predictivos para identificar los comportamientos que preceden al abandono de un cliente.

En cuanto a los algoritmos que pueden utilizarse para este tipo de tareas destacan la regresión logística, los árboles de decisión y el Random Forest y ensambles avanzados como XGboost.

Estos modelos permiten determinar la probabilidad de que un cliente abandone, basándose en factores como la frecuencia de uso del servicio, la antigüedad del cliente, quejas previas y patrones de pago.

Importante: al igual que en la semana anterior, el apunte y el notebook de esta semana sirven como referencias en las que, deliberadamente, no se aplican las técnicas de modelado e interpretación de resultados al caso de negocio (en el caso de esta semana: predicción de churn), con el objetivo de que seas tú mismo(a) quien aplique estas técnicas en el marco de la evaluación, considerando el problema de negocio y los datos que se brindan, los que sí se relacionan con predicción de churn. Lo anterior, dado la calidad de taller del módulo.

1. Consideraciones en la generación de un modelo de predicción de *churn*

1.1. Limpieza de datos

Como en todo modelo debes procurar que tus datos estén limpios, velando por que no haya valores nulos, duplicados, problemas en los valores de tus *strings*, identificación y tratamiento de *outliers* si los algoritmos a utilizar lo ameritan.

1.2. Codificación de variables categóricas

Las **variables categóricas**, como el género, el tipo de plan de suscripción o la ubicación geográfica, no pueden ser interpretadas directamente por los algoritmos de machine learning. Por ello, es necesario transformarlas en representaciones numéricas.

Entre los tipos de codificación más comunes encontramos:

- **Codificación One-Hot:** crea una nueva columna para cada categoría, asignando 1 al valor correspondiente y 0 a los demás. Ejemplo: para una variable Plan con categorías Básico, Estándar y Premium:

Plan_Basico	Plan_Estandar	Plan_Premium
1	0	0
0	1	0
0	0	1

- **Codificación *Label Encoding*:** asigna un número único a cada categoría. Es más simple, pero puede introducir un orden que no existe en los datos.

La recomendación general es utilizar *One-Hot Encoding* cuando las categorías son pocas y no están ordenadas, y *Label Encoding* cuando hay una jerarquía inherente o muchas categorías.

1.3. Métodos de selección de características

Una vez que las variables han sido codificadas, es importante reducir el conjunto de características para mejorar el rendimiento del modelo, evitar el sobreajuste y reducir el tiempo de procesamiento. Existen distintos métodos de selección pero entre los principales encontramos:

- **Importancia de características basada en modelos:** algunos algoritmos, como Random Forest, calculan la importancia de cada característica en el modelo, pudiendo obtener la importancia de cada variable en el modelo (código en notebook de la semana).

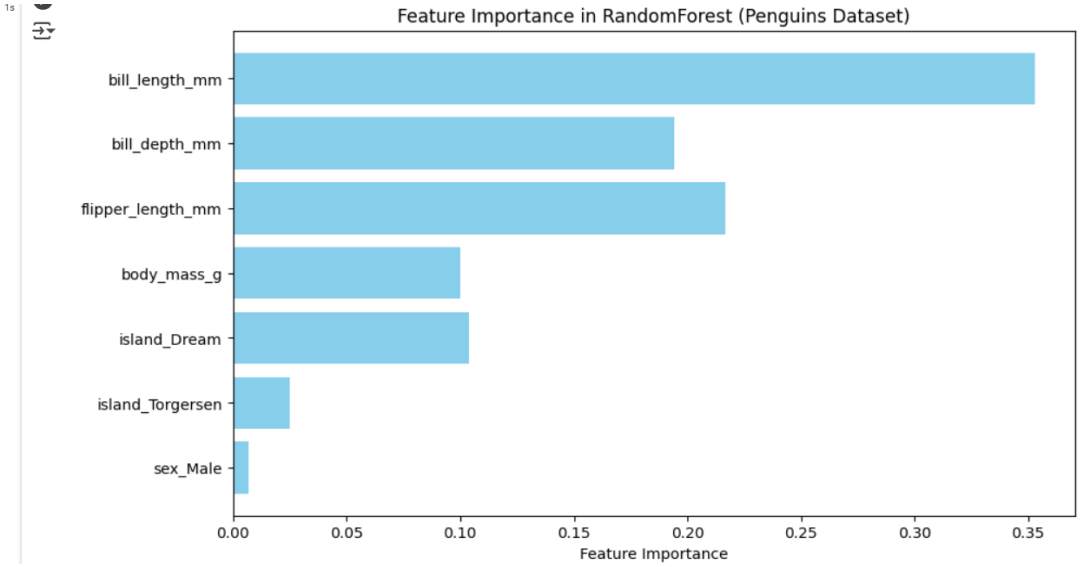


Figura 1. Importancia de características en Random Forest

- **Matriz de correlación:** identifica relaciones fuertes entre variables para eliminar aquellas que aportan información redundante (código en notebook de la semana).

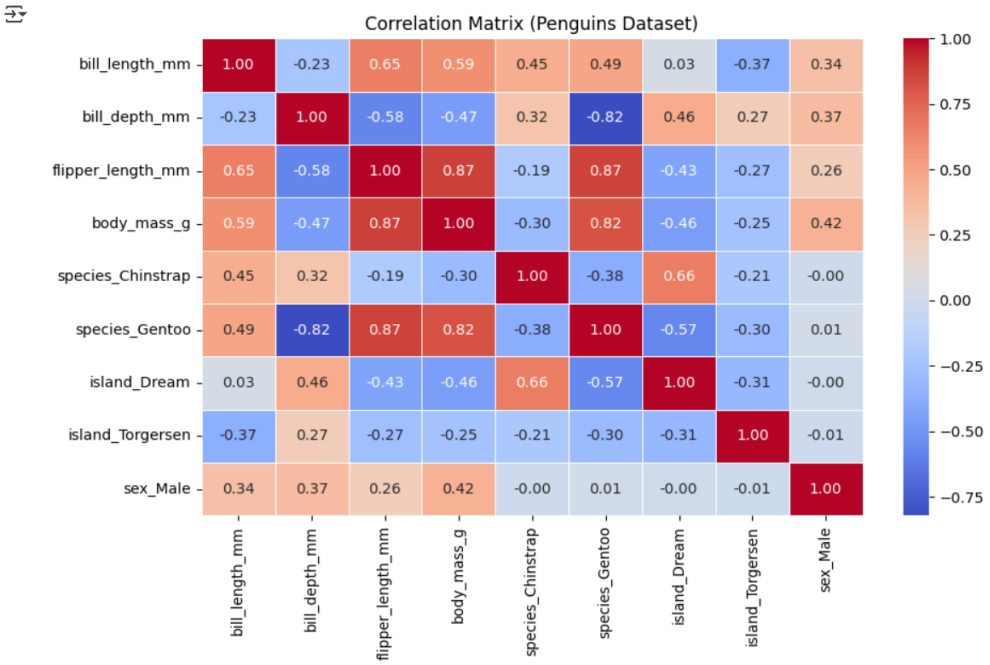


Figura 2. Matriz de correlación

- **Selección automática mediante técnicas de búsqueda:** herramientas como *SelectKBest* eligen automáticamente las mejores características basándose en métricas estadísticas como el chi-cuadrado o la puntuación F:

```
Precisión del modelo tras selección de características: 0.97
      Característica  Importancia
0     bill_length_mm     0.346661
2  flipper_length_mm     0.218088
1      bill_depth_mm     0.194945
3        body_mass_g     0.106277
4       island_Dream     0.104428
5   island_Torgersen     0.023510
6           sex_Male     0.006092
```

Figura 3. Selección automática

Así, en un conjunto de datos optimizado con las características más relevantes, lo que mejora la eficiencia y precisión del modelo predictivo de *churn*.

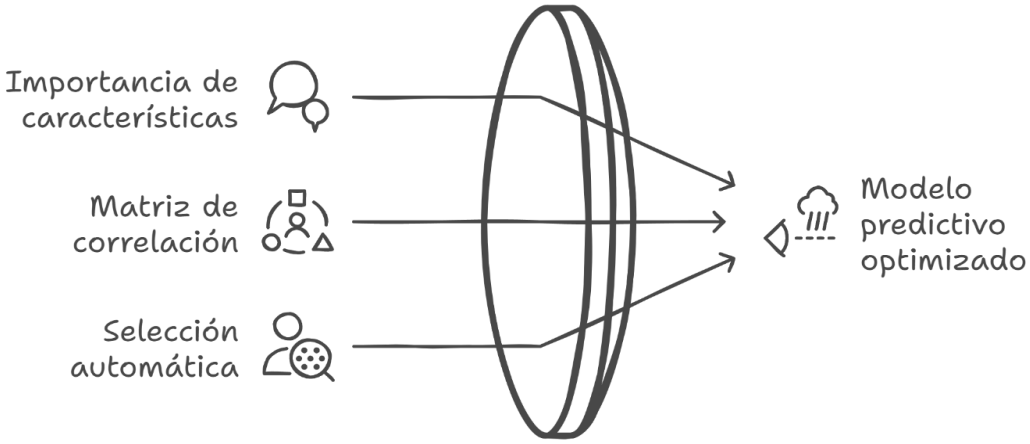


Figura 4. Mejores prácticas en la selección de características. Tres enfoques clave para mejorar los modelos predictivos: importancia de características, matriz de correlación y selección automática, cada uno contribuyendo a la optimización del modelo

1.4. Manejo de desbalanceo de clases

Los conjuntos de datos desbalanceados presentan una distribución desigual entre las clases objetivo (por ejemplo, pocos clientes que abandonan comparados con aquellos que se quedan). Este desbalance puede afectar la precisión del modelo, haciéndolo sesgado hacia la clase mayoritaria.

- ***Oversampling*:** aumenta el número de muestras de la clase minoritaria.
- ***Undersampling*:** reduce el número de muestras de la clase mayoritaria.

```
Original class distribution:
species
Adelie       146
Gentoo       119
Chinstrap     68
Name: count, dtype: int64

Balanced class distribution:
Counter({'Adelie': 146, 'Chinstrap': 146, 'Gentoo': 146})
```

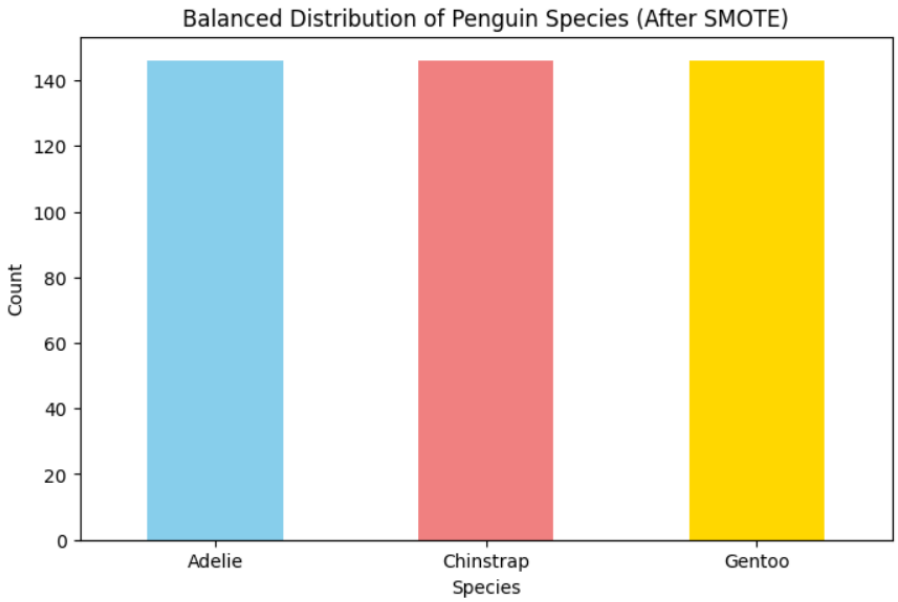


Figura 5. Ejemplo de clases balanceadas

Así, en un conjunto de datos balanceado, con una representación equitativa de todas las clases objetivo, mejora la precisión y la sensibilidad del modelo.

2. Modelamiento de algoritmos clasificadores

Los algoritmos clasificadores son herramientas fundamentales para predecir categorías o clases en conjuntos de datos. En el contexto del análisis de churn, estos algoritmos permiten identificar qué clientes tienen mayor probabilidad de abandonar un servicio. A continuación, se describen cinco algoritmos clave, sus características y cómo implementarlos en Python.

2.1. Regresión logística

La regresión logística es un modelo lineal que predice la probabilidad de que un punto de datos pertenezca a una clase específica. Es útil cuando se requiere una interpretación clara de las relaciones entre las variables. Entre sus ventajas está la fácil interpretación y su eficiencia en problemas linealmente separables; sin embargo, no lo es en problemas no linealmente separables.

2.2. Árboles de decisión

Un árbol de decisión clasifica los datos dividiéndolos en ramas basadas en reglas derivadas de las características. Su resultados son fáciles de interpretar y permiten manejar relaciones no lineales; sin embargo, es un algoritmo propenso al sobreajuste si no se limita su profundidad.

2.3. Random Forest

Es un conjunto de múltiples árboles de decisión. Combina los resultados de cada árbol para mejorar la precisión y reducir el riesgo de sobreajuste. Tiende a tener alta precisión y ser más resistente al sobreajuste que la implementación de un solo árbol; sin embargo, puede ser computacionalmente costoso.

2.4. XGBoost (Extreme Gradient Boosting)

Es un algoritmo de aprendizaje automático basado en árboles de decisión que utiliza la técnica de boosting por gradiente para mejorar el rendimiento del modelo de manera iterativa. A diferencia de Random Forest, que construye múltiples árboles de decisión de manera independiente y combina sus predicciones, XGBoost entrena los árboles de forma secuencial, corrigiendo los errores de los modelos anteriores y ajustando los pesos de las observaciones más difíciles de clasificar. Esto lo hace altamente eficiente y preciso, especialmente en conjuntos de datos estructurados. Además, incorpora mecanismos de regularización para reducir el sobreajuste, lo que lo convierte en una opción más robusta para tareas de clasificación.

3. Evaluación y comparación de modelos: métricas y técnicas de validación

Una vez entrenados los modelos clasificadores, es importante evaluar su rendimiento para garantizar que realicen predicciones precisas y generalicen bien en nuevos datos. Las técnicas de evaluación, como la validación cruzada permiten comparar modelos y seleccionar el más adecuado.

3.1. Validación cruzada (Cross-validation)

La validación cruzada divide los datos en múltiples particiones o pliegues (*folds*) y entrena el modelo en diferentes subconjuntos, evaluando su rendimiento en los datos restantes. Esto reduce el riesgo de sobreajuste. Así, La validación cruzada proporciona una evaluación más confiable, reduciendo la dependencia de un único conjunto de prueba.

3.2. Matriz de confusión

La matriz de confusión muestra la distribución de predicciones correctas e incorrectas para cada clase, lo que ayuda a visualizar el rendimiento del modelo.

	Predicción positiva	**Predicción negativa**
Clase positiva	Verdaderos positivos (VP)	Falsos negativos (FN)
Clase negativa	Falsos positivos (FP)	Verdaderos negativos (VN)

3.3. Métricas de evaluación

Seleccionar una correcta métrica de evaluación y fundamentarla es vital a la hora de evaluar tu modelo de predicción. Dependiendo del problema de negocio, será más importante la sensibilidad (recall), la precisión, el F1, la exactitud o la sensibilidad. Debes saber identificar cuál usarás y por qué.

- **Precisión (Precision):** indica qué proporción de las predicciones positivas es correcta.

$$Precisión = \frac{VP}{VP + FP}$$

- **Recall (Sensibilidad)**: mide la capacidad del modelo para identificar correctamente todas las instancias positivas.

$$Recall = \frac{VP}{VP + FN}$$

- **F1-Score**: es la media armónica entre precisión y recall, útil cuando el balance entre ambas métricas es importante.

$$F1 = 2 \times \frac{Precisión \times Recall}{Precisión + Recall}$$

- **Exactitud (Accuracy)**: proporción de predicciones correctas sobre el total de datos.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

- **Especificidad**: indica la capacidad del modelo para identificar correctamente los negativos.

$$Especificidad = \frac{VN}{VN + FP}$$

Ejemplo (código en notebook de la semana):

```
Optimized Model Performance Summary:

                Model  Accuracy  Precision  Recall  F1-Score     AUC
0  Logistic Regression    0.9333     0.9360  0.9333    0.9332  0.9971
1        Decision Tree    0.9167     0.9224  0.9167    0.9162  0.9583
2        Random Forest    0.9500     0.9507  0.9500    0.9500  0.9733
3              XGBoost    0.8833     0.8854  0.8833    0.8818  0.9467
```

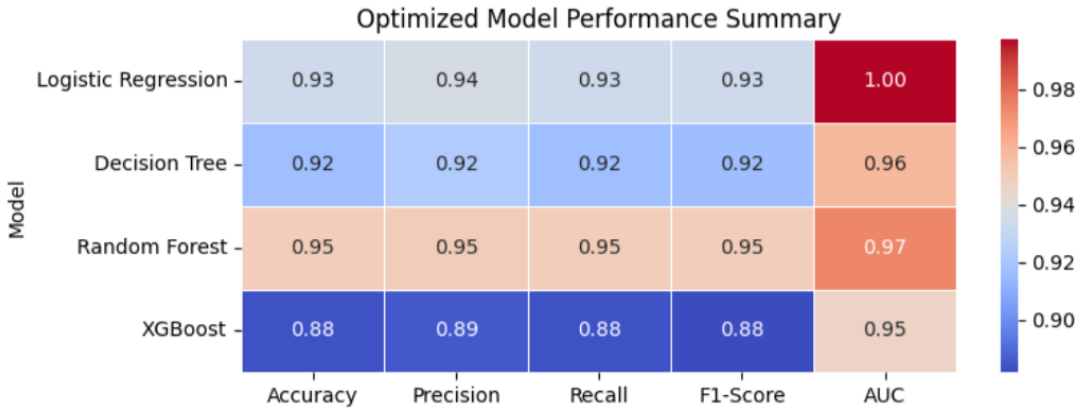


Figura 6. Métricas de evaluación de distintos algoritmos

3.4. Curva ROC y AUC

La **curva ROC (Receiver Operating Characteristic)** es una herramienta fundamental para evaluar el rendimiento de un modelo de clasificación, especialmente en problemas de clasificación binaria. Su principal función es representar gráficamente la relación entre la **tasa de verdaderos positivos (sensibilidad)** y la **tasa de falsos positivos** a distintos umbrales de decisión. Al analizar esta curva, es posible visualizar el

equilibrio entre sensibilidad y especificidad, permitiendo determinar la capacidad del modelo para distinguir entre las clases.

Uno de los indicadores clave extraídos de la curva ROC es el **Área Bajo la Curva (AUC - Area Under the Curve)**, el cual mide la capacidad del modelo para separar correctamente las clases. Un **AUC cercano a 1** indica que el modelo tiene una excelente capacidad de discriminación, mientras que un **AUC de 0.5** sugiere que el modelo no es mejor que una clasificación aleatoria. Este análisis es particularmente útil en conjuntos de datos desbalanceados, ya que la AUC evalúa el rendimiento sin depender de umbrales específicos.

La curva ROC se utiliza para **comparar distintos modelos de clasificación** y seleccionar el más adecuado según el problema a resolver. También permite **ajustar el umbral de decisión**, dependiendo de si se prioriza minimizar los falsos positivos o maximizar la sensibilidad, lo cual es crítico en áreas como la detección de fraudes o enfermedades. En problemas de clasificación **multiclase**, donde hay más de dos categorías, la curva ROC se puede extender utilizando métodos como **One-vs-Rest (OvR)** o **One-vs-One (OvO)**, generando curvas individuales para cada clase y evaluando su rendimiento de manera comparativa.

Ejemplos de curvas ROC para mismo problema con distintos algoritmos (código en notebook de la semana):

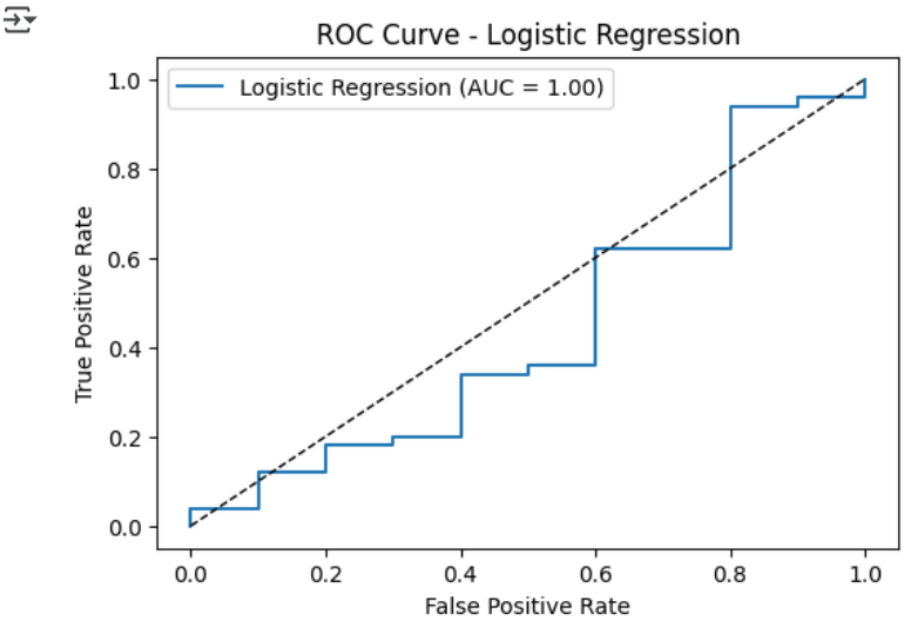


Figura 7. Curva ROC (regresión logística)

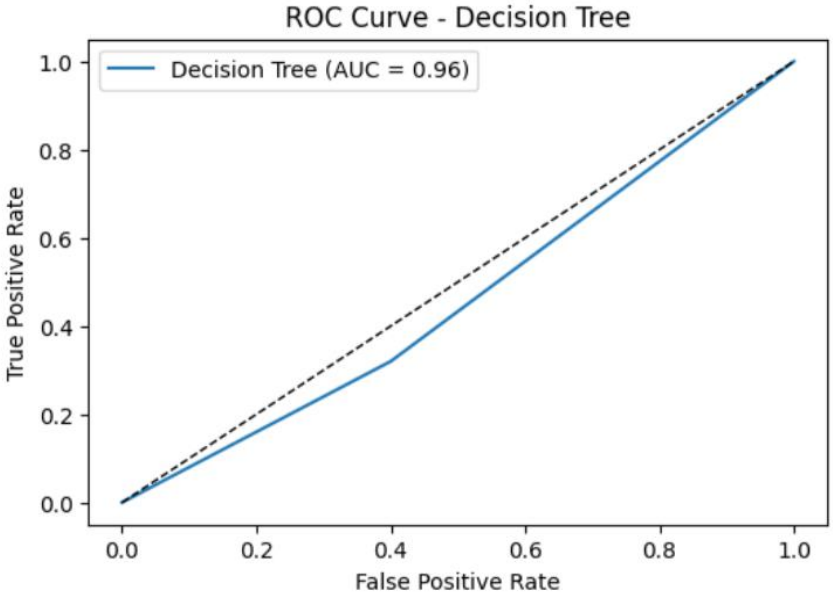


Figura 8. Curva ROC (árbol de decisión)

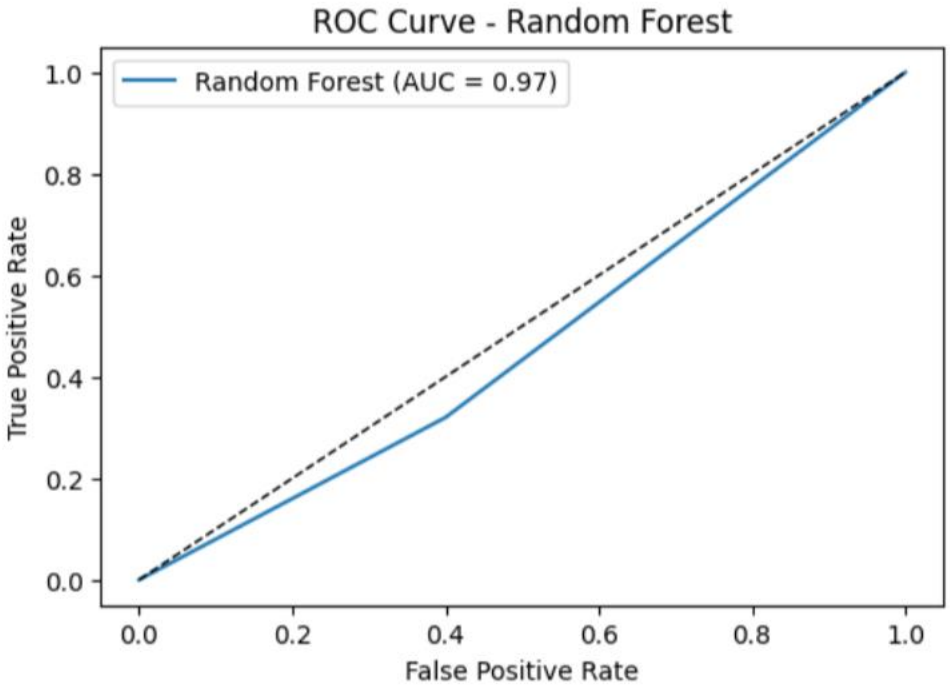


Figura 9. Curva ROC (Random Forest)

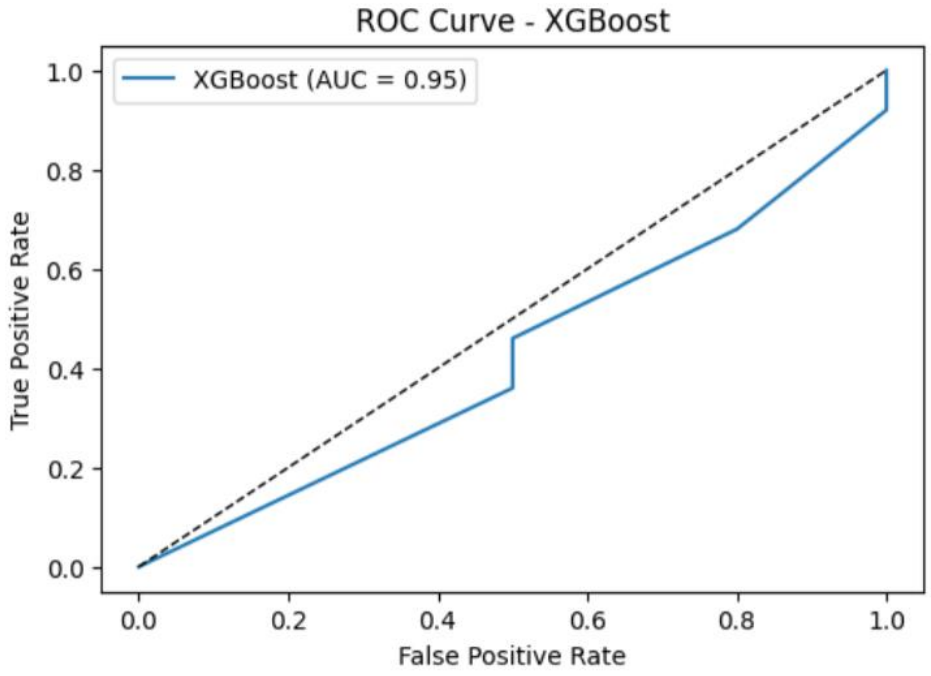


Figura 10. Curva ROC (XGBoost)

4. Interpretación de modelos: comprender el funcionamiento de los clasificadores

Una vez evaluado el modelo, es importante interpretar sus resultados para entender cómo las diferentes características influyen en las predicciones. La interpretación proporciona información valiosa para mejorar las estrategias de negocio o identificar patrones ocultos en los datos.

4.1. Coeficientes y pesos en regresión logística

En la regresión logística, cada característica tiene un coeficiente que indica la fuerza y dirección de su relación con la variable dependiente. Un coeficiente positivo implica que un aumento en esa característica incrementa la probabilidad de pertenecer a la clase positiva, mientras que un coeficiente negativo reduce esa probabilidad. El valor absoluto del coeficiente refleja la magnitud del impacto de la característica. Para calcular el efecto relativo, se puede usar la función logística:

$$odds\ ratio = e^{coeficiente}$$

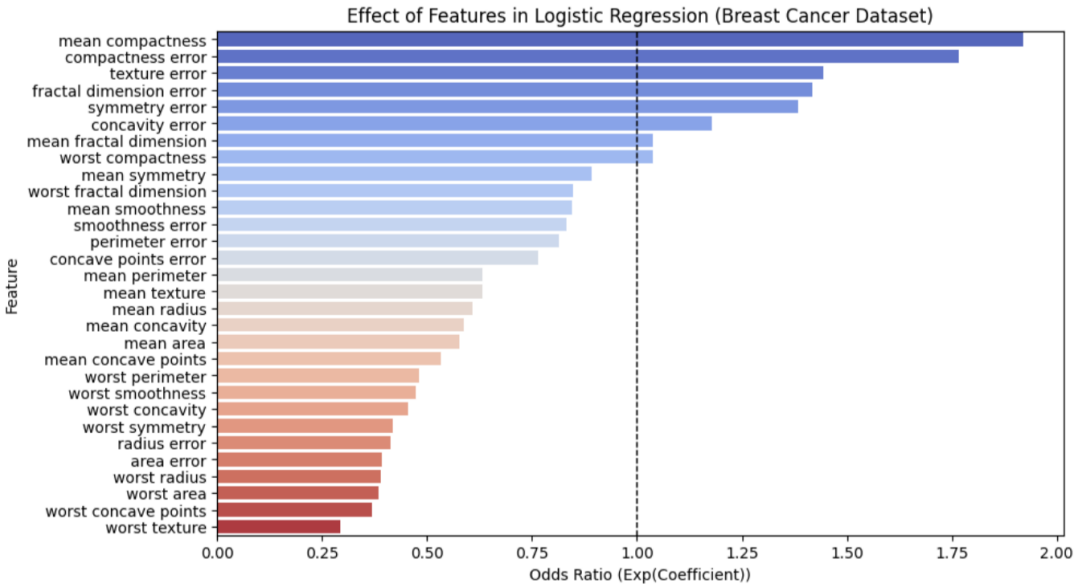


Figura 11. Ejemplo (regresión logística)

4.2. Probabilidad de cada observación de pertenecer a la clase positiva con regresión logística

La regresión logística también nos permite saber la probabilidad de una observación de pertenecer a la clase positiva.

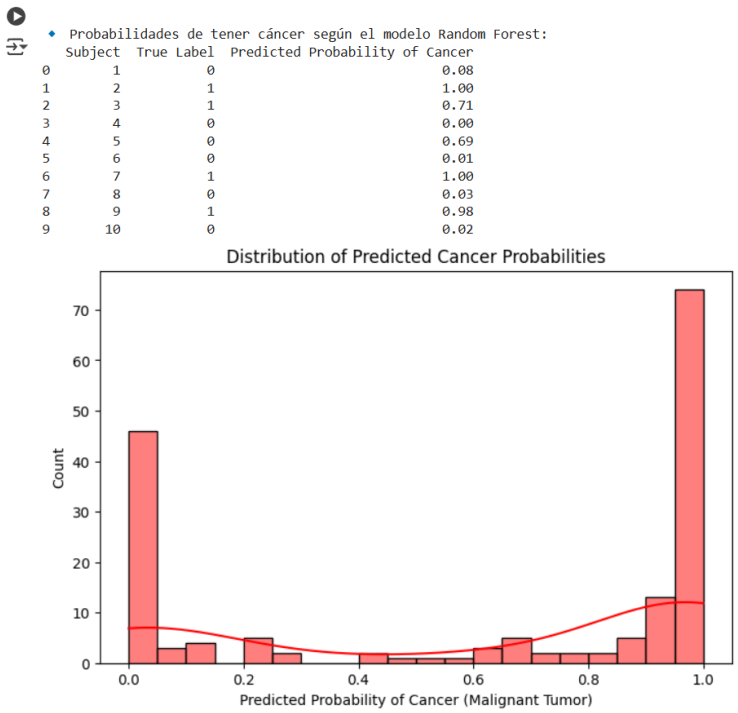


Figura 12. Ejemplo de probabilidad usando Random Forest

4.3. Importancia de características en modelos basados en árboles

Random Forest no permite calcular los Odds Ratio como en Regresión Logística porque estos modelos funcionan de manera completamente diferente. Al usar modelos como árboles de decisión y Random Forest, se calcula la importancia de cada característica en función de su contribución a la reducción de impureza en las divisiones de los nodos. Las características con mayor importancia tienen un papel clave en la toma de decisiones del modelo. Esto puede ayudar a identificar factores críticos para mejorar estrategias empresariales o ajustes de marketing.

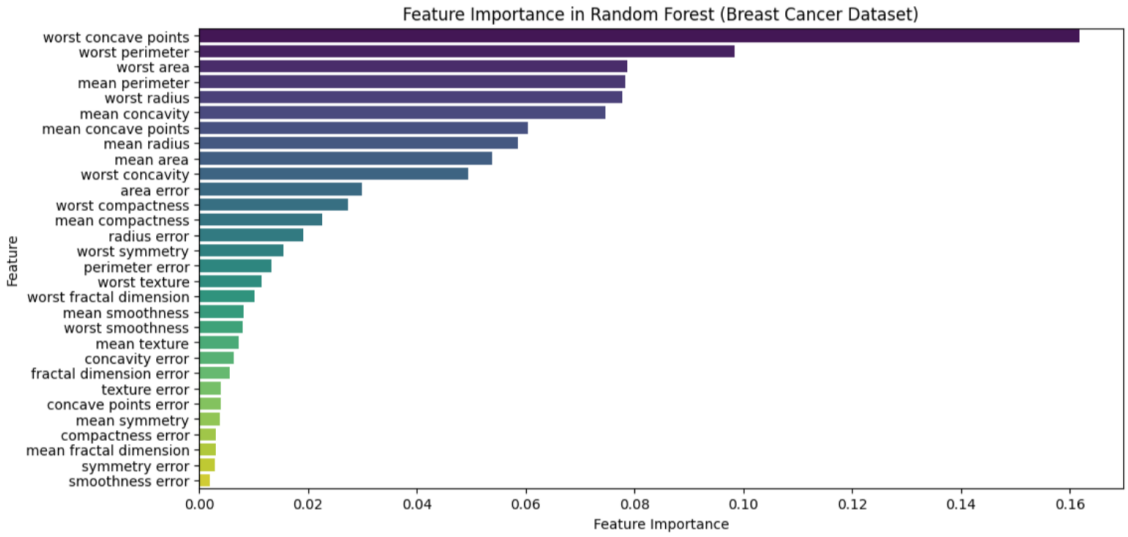


Figura 13. Importancia de características en Random Forest

5. Estrategias para mejorar el modelo: ajuste y optimización

Una vez evaluado e interpretado el modelo, es posible mejorar su rendimiento ajustando los hiperparámetros, que son configuraciones que afectan el comportamiento del algoritmo.

5.1. Ajuste de hiperparámetros

El ajuste de hiperparámetros implica modificar configuraciones como el número de árboles en un Random Forest, el valor de *C* en SVM o la profundidad máxima de un árbol de decisión para encontrar la mejor combinación que maximice el rendimiento.

Ejemplo de hiperparámetros comunes:

- **Árboles de decisión:** max_depth, min_samples_split
- **Random Forest:** n_estimators, max_features
- **SVM:** C, kernel

5.2. Grid Search

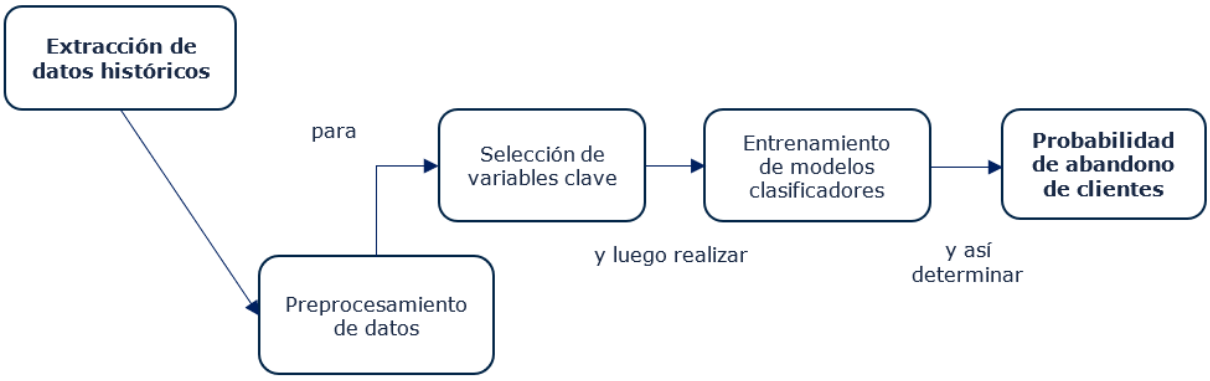
El Grid Search prueba múltiples combinaciones de hiperparámetros de manera sistemática, evaluando cada combinación con validación cruzada para seleccionar la mejor configuración.

Un modelo optimizado con los mejores hiperparámetros, lo que mejora la precisión, generalización y rendimiento en datos de prueba. La combinación

de ajuste manual y técnicas automatizadas como Grid Search permite mejorar los resultados de manera eficiente.

Cierre

Por medio del siguiente organizador gráfico se destacan las ideas clave de esta semana:



La interpretación adecuada de los modelos y la optimización de sus parámetros son pasos esenciales para mejorar su rendimiento y aplicabilidad en escenarios reales. Los coeficientes en la regresión logística permiten identificar la influencia de cada variable en la predicción, mientras que los modelos basados en árboles destacan las características más relevantes mediante la reducción de impureza. Al aplicar estrategias como el ajuste de hiperparámetros y el Grid Search, es posible mejorar significativamente la capacidad del modelo para generalizar en datos no vistos, obteniendo así resultados más precisos y confiables.

Referencias

Envato Elements (s. f.). Visión desde un ángulo elevado de los trabajadores. [Imagen]. https://elements.envato.com/es/high-angle-view-on-working-people-VK423DJ

(s. f.) Un hombre sostiene una caja de cartón y está de pie frente a una pizarra blanca. Está haciendo una presentación y sostiene la caja para mostrar algo [Imagen]. https://elements.envato.com/es/group-of-young-male-and-female-business-people-in--EUNHC8M

Las figuras (diagramas e imágenes) utilizadas en este apunte fueron generadas mediante Napkin (https://www.napkin.ai/) (figura 4) y el notebook asociado al módulo (demás figuras).