

Asesoría Técnica Final: Análisis Exhaustivo del Notebook de Predicción de Customer Churn

Equipo de Data Science

Contents

| | |
|---|----------|
| 1 Asesoría Técnica Final: Análisis Exhaustivo del Notebook de Predicción de Customer Churn | 3 |
| 1.1 Estructura General del Notebook | 3 |
| 1.2 Bloque 0: Importación de Librerías | 3 |
| 1.2.1 ¿Qué se busca? | 3 |
| 1.2.2 ¿Por qué se aplica? | 4 |
| 1.2.3 ¿Qué aporta al notebook? | 4 |
| 1.2.4 Glosario contextualizado | 4 |
| 1.3 Bloque 0.1: Configuración de Reproducibilidad | 4 |
| 1.3.1 ¿Qué se busca? | 4 |
| 1.3.2 ¿Por qué se aplica? | 4 |
| 1.3.3 ¿Qué aporta al notebook? | 5 |
| 1.3.4 Glosario contextualizado | 5 |
| 1.4 Bloque 1: Carga y Exploración Inicial de Datos | 5 |
| 1.4.1 ¿Qué se busca? | 5 |
| 1.4.2 ¿Por qué se aplica? | 5 |
| 1.4.3 ¿Qué aporta al notebook? | 5 |
| 1.4.4 Glosario contextualizado | 6 |
| 1.5 Bloque 2: Análisis de Calidad de Datos | 6 |
| 1.5.1 ¿Qué se busca? | 6 |
| 1.5.2 ¿Por qué se aplica? | 6 |
| 1.5.3 ¿Qué aporta al notebook? | 6 |
| 1.5.4 Glosario contextualizado | 6 |
| 1.6 Bloque 3: Análisis Exploratorio de Datos (EDA) | 7 |
| 1.6.1 ¿Qué se busca? | 7 |
| 1.6.2 ¿Por qué se aplica? | 7 |
| 1.6.3 ¿Qué aporta al notebook? | 7 |

| | | |
|--------|---|----|
| 1.6.4 | Glosario contextualizado | 7 |
| 1.7 | Bloque 4: Feature Engineering | 8 |
| 1.7.1 | ¿Qué se busca? | 8 |
| 1.7.2 | ¿Por qué se aplica? | 8 |
| 1.7.3 | ¿Qué aporta al notebook? | 8 |
| 1.7.4 | Glosario contextualizado | 8 |
| 1.8 | Bloque 5: Preparación de Datos para Modelado | 8 |
| 1.8.1 | ¿Qué se busca? | 8 |
| 1.8.2 | ¿Por qué se aplica? | 9 |
| 1.8.3 | ¿Qué aporta al notebook? | 9 |
| 1.8.4 | Glosario contextualizado | 9 |
| 1.9 | Bloque 6: Entrenamiento de Modelos Baseline | 9 |
| 1.9.1 | ¿Qué se busca? | 9 |
| 1.9.2 | ¿Por qué se aplica? | 9 |
| 1.9.3 | ¿Qué aporta al notebook? | 10 |
| 1.9.4 | Glosario contextualizado | 10 |
| 1.10 | Bloque 7: Manejo del Desbalanceo de Clases | 10 |
| 1.10.1 | ¿Qué se busca? | 10 |
| 1.10.2 | ¿Por qué se aplica? | 10 |
| 1.10.3 | ¿Qué aporta al notebook? | 11 |
| 1.10.4 | Glosario contextualizado | 11 |
| 1.11 | Bloque 8: Optimización de Hiperparámetros | 11 |
| 1.11.1 | ¿Qué se busca? | 11 |
| 1.11.2 | ¿Por qué se aplica? | 11 |
| 1.11.3 | ¿Qué aporta al notebook? | 11 |
| 1.11.4 | Glosario contextualizado | 12 |
| 1.12 | Bloque 9: Evaluación Detallada del Mejor Modelo | 12 |
| 1.12.1 | ¿Qué se busca? | 12 |
| 1.12.2 | ¿Por qué se aplica? | 12 |
| 1.12.3 | ¿Qué aporta al notebook? | 12 |
| 1.12.4 | Glosario contextualizado | 13 |
| 1.13 | Bloque 10: Conclusiones y Recomendaciones (Generadas Dinámicamente) | 13 |
| 1.13.1 | ¿Qué se busca? | 13 |
| 1.13.2 | ¿Por qué se aplica? | 13 |
| 1.13.3 | ¿Qué aporta al notebook? | 14 |
| 1.13.4 | Glosario contextualizado | 14 |

| | | |
|--------|---|----|
| 1.14 | Bloque 11: Guardar el Modelo y Verificar Tamaño para Deployment | 14 |
| 1.14.1 | ¿Qué se busca? | 14 |
| 1.14.2 | ¿Por qué se aplica? | 14 |
| 1.14.3 | ¿Qué aporta al notebook? | 14 |
| 1.14.4 | Glosario contextualizado | 15 |
| 1.15 | Bloque 12: Resumen Técnico del Proyecto | 15 |
| 1.15.1 | ¿Qué se busca? | 15 |
| 1.15.2 | ¿Por qué se aplica? | 15 |
| 1.15.3 | ¿Qué aporta al notebook? | 15 |
| 1.15.4 | Glosario contextualizado | 16 |
| 1.16 | Bloque 13: Generación de Informe Automático | 16 |
| 1.16.1 | ¿Qué se busca? | 16 |
| 1.16.2 | ¿Por qué se aplica? | 16 |
| 1.16.3 | ¿Qué aporta al notebook? | 16 |
| 1.16.4 | Glosario contextualizado | 17 |
| 1.17 | Conclusión General del Análisis | 17 |
| 1.17.1 | Fortalezas del Proyecto | 17 |
| 1.17.2 | Aplicabilidad en el Contexto de Telecomunicaciones | 17 |
| 1.17.3 | Recomendaciones para Mejora Continua | 18 |

1 Asesoría Técnica Final: Análisis Exhaustivo del Notebook de Predicción de Customer Churn

Proyecto: Telco Customer Churn Prediction

Archivo: Telco_Customer_Churn.ipynb

Fecha de Análisis: 2025-11-23

Objetivo: Desarrollar modelos de Machine Learning para identificar clientes con alta probabilidad de abandonar el servicio

1.1 Estructura General del Notebook

Este notebook implementa un pipeline completo de Machine Learning para predicción de churn, organizado en 13 bloques principales que siguen una metodología estructurada desde la carga de datos hasta el deployment del modelo.

1.2 Bloque 0: Importación de Librerías

1.2.1 ¿Qué se busca?

Importar y configurar todas las dependencias necesarias para el proyecto, incluyendo librerías para manipulación de datos, visualización, preprocesamiento, modelado, evaluación y manejo de desbalanceo de clases.

1.2.2 ¿Por qué se aplica?

Es fundamental establecer el entorno de trabajo completo al inicio del notebook para:

- Garantizar que todas las herramientas estén disponibles desde el principio
- Evitar errores de importación durante la ejecución
- Configurar parámetros globales de visualización de manera consistente
- Suprimir advertencias innecesarias que puedan distraer del análisis

1.2.3 ¿Qué aporta al notebook?

- **Ecosistema completo de herramientas:** Desde pandas/numpy para datos hasta XGBoost para modelado avanzado
- **Configuración visual estandarizada:** Estilo seaborn, paleta de colores y tamaño de figuras pre-definidos
- **Supresión de warnings:** Entorno limpio para análisis sin distracciones
- **Preparación para SMOTE:** Importación de imblearn para manejar desbalanceo de clases

1.2.4 Glosario contextualizado

- **pandas:** Librería para manipulación y análisis de datos tabulares (DataFrames)
 - **numpy:** Librería para operaciones numéricas y manejo de arrays
 - **matplotlib/seaborn:** Librerías de visualización de datos
 - **sklearn:** Scikit-learn, framework principal para Machine Learning
 - **StandardScaler:** Normalizador que transforma variables a media 0 y desviación estándar 1
 - **LabelEncoder:** Codificador que convierte variables categóricas en numéricas
 - **ColumnTransformer:** Permite aplicar diferentes transformaciones a diferentes columnas
 - **Pipeline:** Encadena múltiples pasos de preprocesamiento y modelado
 - **SMOTE:** Synthetic Minority Over-sampling Technique, técnica para balancear clases desbalanceadas
 - **RandomizedSearchCV:** Búsqueda aleatoria de hiperparámetros óptimos
 - **ROC-AUC:** Área bajo la curva ROC, métrica de capacidad discriminativa del modelo
 - **confusion_matrix:** Matriz que muestra verdaderos/falsos positivos y negativos
 - **XGBoost:** Extreme Gradient Boosting, algoritmo avanzado de ensemble learning
-

1.3 Bloque 0.1: Configuración de Reproducibilidad

1.3.1 ¿Qué se busca?

Establecer un sistema de control de semillas aleatorias que permita alternar entre dos modos de ejecución: reproducible (resultados idénticos) y experimental (resultados variables).

1.3.2 ¿Por qué se aplica?

En proyectos de Machine Learning, la aleatoriedad afecta múltiples procesos (división de datos, inicialización de modelos, SMOTE, búsqueda de hiperparámetros). Este bloque permite:

- **Modo reproducible:** Para documentación, presentaciones y validación de resultados
- **Modo experimental:** Para probar la robustez del modelo con diferentes semillas
- **Transparencia:** El usuario sabe exactamente qué semilla se está usando

1.3.3 ¿Qué aporta al notebook?

- **Control total sobre la aleatoriedad:** Variable global `RANDOM_STATE` usada en todo el notebook
- **Flexibilidad:** Cambio rápido entre modos con un solo parámetro (`REPRODUCIBLE_MODE`)
- **Trazabilidad:** Imprime la semilla utilizada para documentar cada ejecución
- **Experimentación científica:** Permite validar que el modelo es robusto ante diferentes particiones de datos

1.3.4 Glosario contextualizado

- **Semilla aleatoria (random seed):** Valor inicial para generadores de números pseudoaleatorios que garantiza reproducibilidad
 - **`RANDOM_STATE`:** Variable global que almacena la semilla usada en todos los procesos aleatorios
 - **Reproducibilidad:** Capacidad de obtener los mismos resultados en ejecuciones repetidas
 - **Modo experimental:** Configuración que usa semillas diferentes en cada ejecución para probar robustez
 - **`np.random.randint()`:** Función de NumPy que genera un entero aleatorio en un rango especificado
-

1.4 Bloque 1: Carga y Exploración Inicial de Datos

1.4.1 ¿Qué se busca?

Cargar el dataset desde diferentes ubicaciones posibles (local, Google Drive) y realizar una primera inspección de su estructura, dimensiones y contenido.

1.4.2 ¿Por qué se aplica?

Este bloque es crítico porque:

- Establece la base de datos sobre la cual se trabajará todo el análisis
- Implementa un sistema robusto de carga que funciona en múltiples entornos (local, Colab)
- Proporciona una primera visión de la calidad y estructura de los datos
- Identifica el número de registros y variables disponibles

1.4.3 ¿Qué aporta al notebook?

- **Función robusta de carga:** Intenta múltiples rutas posibles, incluyendo Google Drive
- **Montaje automático de Drive:** Facilita el trabajo en Google Colab
- **Información básica del dataset:** 7,043 clientes con 21 variables
- **Primera inspección visual:** Muestra las primeras filas para entender la estructura
- **Tipos de datos:** Identifica variables numéricas (`tenure`, `MonthlyCharges`) y categóricas (`gender`, `Contract`)
- **Estadísticas descriptivas:** Media, desviación estándar, cuartiles de variables numéricas

1.4.4 Glosario contextualizado

- **DataFrame:** Estructura de datos tabular de pandas similar a una hoja de cálculo
 - **customerID:** Identificador único de cada cliente
 - **tenure:** Número de meses que el cliente ha permanecido con la empresa
 - **MonthlyCharges:** Cargo mensual actual del cliente
 - **TotalCharges:** Suma total de cargos acumulados del cliente
 - **Churn:** Variable objetivo (Yes/No) que indica si el cliente abandonó el servicio
 - **SeniorCitizen:** Variable binaria (0/1) que indica si el cliente es adulto mayor
 - **Contract:** Tipo de contrato (Month-to-month, One year, Two year)
 - **InternetService:** Tipo de servicio de internet (DSL, Fiber optic, No)
 - **df.shape:** Tupla que indica (número de filas, número de columnas)
 - **df.info():** Método que muestra tipos de datos y valores no nulos por columna
 - **df.describe():** Genera estadísticas descriptivas de variables numéricas
-

1.5 Bloque 2: Análisis de Calidad de Datos

1.5.1 ¿Qué se busca?

Detectar y corregir problemas de calidad en los datos, específicamente valores faltantes, tipos de datos incorrectos y anomalías en la variable TotalCharges.

1.5.2 ¿Por qué se aplica?

La calidad de los datos determina la calidad del modelo. Este bloque es esencial porque:

- Identifica que TotalCharges está almacenado como texto (object) en lugar de numérico
- Detecta 11 registros con espacios en blanco en TotalCharges
- Todos estos registros corresponden a clientes nuevos (tenure=0)
- Los valores faltantes pueden causar errores en el modelado

1.5.3 ¿Qué aporta al notebook?

- **Detección de anomalías:** Identifica que TotalCharges tiene espacios en blanco
- **Conversión de tipos:** Transforma TotalCharges de object a float64
- **Imputación inteligente:** Para clientes con tenure=0, TotalCharges = MonthlyCharges
- **Dataset limpio:** 0 valores faltantes después del procesamiento
- **Validación:** Confirma que todos los registros están completos y listos para modelado

1.5.4 Glosario contextualizado

- **Valores faltantes (missing values):** Datos ausentes en el dataset (NaN, None, espacios)
- **object dtype:** Tipo de dato de pandas para texto/cadenas
- **float64:** Tipo de dato numérico de punto flotante de 64 bits
- **pd.to_numeric():** Función que convierte valores a numérico, con manejo de errores
- **errors='coerce':** Parámetro que convierte valores no convertibles a NaN
- **Imputación:** Proceso de rellenar valores faltantes con valores estimados
- **df.isnull().sum():** Cuenta valores nulos por columna

- **df.loc[]**: Indexador de pandas para seleccionar filas y columnas por etiquetas
 - **Anomalía**: Valor inusual o inesperado en los datos que requiere investigación
-

1.6 Bloque 3: Análisis Exploratorio de Datos (EDA)

1.6.1 ¿Qué se busca?

Comprender profundamente la distribución de las variables, sus relaciones con el churn y los patrones ocultos en los datos mediante visualizaciones y análisis estadísticos.

1.6.2 ¿Por qué se aplica?

El EDA es fundamental para:

- Identificar qué variables están más relacionadas con el churn
- Detectar desbalanceo en la variable objetivo (73% No Churn vs 27% Churn)
- Descubrir patrones de comportamiento de clientes que abandonan
- Guiar la selección de características y estrategias de modelado
- Validar supuestos sobre los datos

1.6.3 ¿Qué aporta al notebook?

- **Análisis de desbalanceo**: Identifica que solo 26.5% de clientes hacen churn
- **Visualizaciones categóricas**: Gráficos de barras mostrando churn por género, tipo de contrato, servicios
- **Insights clave**:
 - Contratos mes a mes tienen mayor churn
 - Clientes con Fiber optic tienen mayor churn
 - Clientes sin servicios adicionales (seguridad, backup) tienen mayor churn
- **Análisis numérico**: Distribuciones de tenure, MonthlyCharges, TotalCharges
- **Correlaciones**: Matriz de correlación entre variables numéricas
- **Patrones temporales**: Clientes nuevos (tenure bajo) tienen mayor probabilidad de churn

1.6.4 Glosario contextualizado

- **EDA (Exploratory Data Analysis)**: Análisis exploratorio para entender patrones en los datos
 - **Desbalanceo de clases**: Cuando una clase tiene significativamente más ejemplos que otra
 - **Variable objetivo (target)**: Variable que queremos predecir (Churn)
 - **Variables predictoras (features)**: Variables usadas para hacer la predicción
 - **Distribución**: Patrón de frecuencia de valores en una variable
 - **Correlación**: Medida de relación lineal entre dos variables (-1 a 1)
 - **Heatmap**: Mapa de calor que visualiza correlaciones con colores
 - **Countplot**: Gráfico de barras que muestra frecuencias de categorías
 - **Histograma**: Gráfico que muestra la distribución de una variable continua
 - **Boxplot**: Gráfico de caja que muestra mediana, cuartiles y outliers
 - **Insight**: Descubrimiento o patrón relevante encontrado en los datos
-

1.7 Bloque 4: Feature Engineering

1.7.1 ¿Qué se busca?

Crear nuevas características derivadas de las variables existentes que puedan capturar patrones más complejos y mejorar el poder predictivo de los modelos.

1.7.2 ¿Por qué se aplica?

El feature engineering es crucial porque:

- Los modelos pueden no capturar automáticamente relaciones complejas entre variables
- Características derivadas pueden revelar patrones ocultos relacionados con el churn
- Permite incorporar conocimiento del dominio del negocio
- Puede mejorar significativamente el rendimiento del modelo

1.7.3 ¿Qué aporta al notebook?

Crea 6 nuevas características estratégicas:

1. **ChargeRatio**: Relación entre cargo mensual y total (detecta clientes nuevos vs antiguos)
2. **AvgMonthlyCharges**: Promedio real de cargos mensuales basado en tenure
3. **TenureGroup**: Categorización de tenure en grupos (0-1 año, 1-2 años, 2-4 años, 4+ años)
4. **TotalServices**: Contador de servicios contratados (de 0 a 8)
5. **SeniorWithDependents**: Interacción entre ser senior y tener dependientes
6. **HighValueContract**: Identifica contratos largos con cargos altos

1.7.4 Glosario contextualizado

- **Feature Engineering**: Proceso de crear nuevas variables a partir de las existentes
 - **Características derivadas**: Variables calculadas combinando o transformando variables originales
 - **ChargeRatio**: Proporción que indica qué tan nuevo es el cliente (valores altos = cliente reciente)
 - **pd.cut()**: Función que divide una variable continua en intervalos categóricos
 - **bins**: Límites de los intervalos para categorización
 - **labels**: Etiquetas asignadas a cada intervalo
 - **Interacción de variables**: Combinación de dos o más variables para capturar efectos conjuntos
 - **Variable binaria**: Variable que solo toma valores 0 o 1
 - **astype(int)**: Conversión de tipo booleano a entero
-

1.8 Bloque 5: Preparación de Datos para Modelado

1.8.1 ¿Qué se busca?

Transformar el dataset en un formato adecuado para algoritmos de Machine Learning mediante codificación de variables categóricas, escalado de numéricas y división en conjuntos de entrenamiento y prueba.

1.8.2 ¿Por qué se aplica?

Los algoritmos de ML requieren:

- Variables numéricas (no pueden procesar texto directamente)
- Escalas similares entre variables (para algoritmos sensibles a magnitudes)
- Separación de datos para evaluar el modelo en datos no vistos
- Preservación de la proporción de clases en train/test (stratify)

1.8.3 ¿Qué aporta al notebook?

- **Codificación de variables categóricas:** Convierte Yes/No, tipos de contrato, etc. a números
- **Eliminación de variables irrelevantes:** Descarta customerID y TenureGroup
- **División estratificada:** 80% entrenamiento, 20% prueba manteniendo proporción de churn
- **Pipeline de preprocesamiento:**
 - One-Hot Encoding para categóricas
 - StandardScaler para numéricas
- **Datos listos para modelado:** X_train, X_test, y_train, y_test procesados

1.8.4 Glosario contextualizado

- **Codificación (encoding):** Conversión de variables categóricas a numéricas
 - **LabelEncoder:** Asigna un número único a cada categoría (0, 1, 2, ...)
 - **One-Hot Encoding:** Crea una columna binaria por cada categoría
 - **train_test_split:** Función que divide datos en entrenamiento y prueba
 - **test_size=0.2:** 20% de datos para prueba, 80% para entrenamiento
 - **stratify:** Mantiene la misma proporción de clases en train y test
 - **StandardScaler:** Normaliza variables a media 0 y desviación estándar 1
 - **ColumnTransformer:** Aplica diferentes transformaciones a diferentes columnas
 - **Pipeline:** Secuencia de transformaciones aplicadas en orden
 - **fit_transform():** Aprende parámetros y transforma datos de entrenamiento
 - **transform():** Aplica transformación aprendida a datos de prueba
-

1.9 Bloque 6: Entrenamiento de Modelos Baseline

1.9.1 ¿Qué se busca?

Entrenar múltiples algoritmos de Machine Learning para establecer una línea base de rendimiento y identificar qué tipos de modelos funcionan mejor para este problema.

1.9.2 ¿Por qué se aplica?

Es una buena práctica probar varios algoritmos porque:

- Diferentes algoritmos capturan diferentes patrones en los datos
- No se puede saber a priori cuál funcionará mejor
- Permite comparar rendimiento y seleccionar el mejor candidato
- Establece un benchmark para mejoras posteriores

1.9.3 ¿Qué aporta al notebook?

Entrena y evalúa 7 modelos diferentes:

1. **Logistic Regression**: Modelo lineal simple (ROC-AUC: 0.849)
2. **Decision Tree**: Árbol de decisión (ROC-AUC: 0.628)
3. **Random Forest**: Ensemble de árboles (ROC-AUC: 0.818)
4. **Gradient Boosting**: Boosting secuencial (ROC-AUC: 0.846)
5. **XGBoost**: Gradient boosting optimizado (ROC-AUC: 0.821)
6. **SVM**: Support Vector Machine (ROC-AUC: 0.797)
7. **KNN**: K-Nearest Neighbors (ROC-AUC: 0.795)

Resultado: Logistic Regression obtiene el mejor ROC-AUC baseline (0.849)

1.9.4 Glosario contextualizado

- **Modelo baseline**: Modelo inicial sin optimización que sirve como referencia
 - **Logistic Regression**: Modelo lineal para clasificación binaria
 - **Decision Tree**: Árbol que divide datos según reglas de decisión
 - **Random Forest**: Conjunto de árboles de decisión que votan
 - **Gradient Boosting**: Construye árboles secuencialmente corrigiendo errores previos
 - **XGBoost**: Implementación optimizada de gradient boosting
 - **SVM (Support Vector Machine)**: Encuentra hiperplano óptimo de separación
 - **KNN (K-Nearest Neighbors)**: Clasifica según vecinos más cercanos
 - **Accuracy**: Porcentaje de predicciones correctas
 - **Precision**: De los predichos como churn, cuántos realmente lo son
 - **Recall**: De los que realmente hacen churn, cuántos detectamos
 - **F1-Score**: Media armónica de precision y recall
 - **ROC-AUC**: Área bajo curva ROC, mide capacidad discriminativa (0.5-1.0)
-

1.10 Bloque 7: Manejo del Desbalanceo de Clases

1.10.1 ¿Qué se busca?

Balancear el dataset aplicando SMOTE (Synthetic Minority Over-sampling Technique) para generar ejemplos sintéticos de la clase minoritaria (Churn=Yes) y mejorar la capacidad del modelo para detectar churn.

1.10.2 ¿Por qué se aplica?

El desbalanceo de clases (73% No Churn vs 27% Churn) causa problemas:

- Los modelos tienden a predecir la clase mayoritaria
- Baja capacidad para detectar churn (bajo recall)
- El modelo aprende más sobre clientes que se quedan que sobre los que se van
- SMOTE genera ejemplos sintéticos inteligentes (no duplicados) de la clase minoritaria

1.10.3 ¿Qué aporta al notebook?

- **Balanceo perfecto:** De ratio 2.77:1 a 1.00:1 (4,139 ejemplos de cada clase)
- **Datos sintéticos inteligentes:** SMOTE crea ejemplos interpolando entre vecinos cercanos
- **Mejora en recall:** Los modelos reentrenados detectan mejor el churn
- **Comparación antes/después:**
 - Logistic Regression: Recall sube de 0.535 a 0.781
 - Random Forest: Recall sube de 0.476 a 0.570
- **Preservación del conjunto de prueba:** SMOTE solo se aplica a entrenamiento

1.10.4 Glosario contextualizado

- **SMOTE:** Técnica que crea ejemplos sintéticos de la clase minoritaria
 - **Oversampling:** Aumentar el número de ejemplos de la clase minoritaria
 - **Undersampling:** Reducir el número de ejemplos de la clase mayoritaria
 - **Clase minoritaria:** La clase con menos ejemplos (Churn=Yes, 27%)
 - **Clase mayoritaria:** La clase con más ejemplos (Churn=No, 73%)
 - **Ejemplos sintéticos:** Nuevos datos generados artificialmente, no duplicados
 - **fit_resample():** Método de SMOTE que genera los ejemplos sintéticos
 - **Ratio de clases:** Proporción entre clase mayoritaria y minoritaria
 - **Trade-off precision-recall:** Al mejorar recall, puede bajar precision (más falsos positivos)
-

1.11 Bloque 8: Optimización de Hiperparámetros

1.11.1 ¿Qué se busca?

Encontrar la mejor combinación de hiperparámetros para Random Forest mediante búsqueda aleatoria (RandomizedSearchCV) que maximice el ROC-AUC en validación cruzada.

1.11.2 ¿Por qué se aplica?

Los hiperparámetros por defecto rara vez son óptimos:

- Cada dataset requiere configuración específica
- La búsqueda sistemática encuentra mejores configuraciones
- RandomizedSearchCV es más eficiente que GridSearchCV (prueba combinaciones aleatorias)
- La validación cruzada asegura que el modelo generalice bien

1.11.3 ¿Qué aporta al notebook?

- **Búsqueda optimizada:** 20 iteraciones con 3 folds (vs 50 iteraciones y 5 folds original)
- **Reducción de tiempo:** De ~20 minutos a ~3 minutos (85% más rápido)
- **Hiperparámetros explorados:**
 - n_estimators: [100, 200, 300]
 - max_depth: [10, 20, None]
 - min_samples_split: [2, 5]
 - min_samples_leaf: [1, 2]

- max_features: ['sqrt', 'log2']
- bootstrap: [True, False]
- **Mejor configuración encontrada:** Varía según la semilla aleatoria
- **ROC-AUC en validación cruzada:** ~0.936 (excelente)
- **ROC-AUC en test:** ~0.827

1.11.4 Glosario contextualizado

- **Hiperparámetros:** Parámetros del modelo que se configuran antes del entrenamiento
- **RandomizedSearchCV:** Búsqueda aleatoria de hiperparámetros con validación cruzada
- **GridSearchCV:** Búsqueda exhaustiva de todas las combinaciones posibles
- **n_estimators:** Número de árboles en el Random Forest
- **max_depth:** Profundidad máxima de cada árbol
- **min_samples_split:** Mínimo de muestras para dividir un nodo
- **min_samples_leaf:** Mínimo de muestras en una hoja
- **max_features:** Número de características a considerar en cada división
- **bootstrap:** Si se usa muestreo con reemplazo para crear árboles
- **Validación cruzada (CV):** Divide datos en k partes, entrena k veces
- **n_iter:** Número de combinaciones aleatorias a probar
- **cv=3:** Validación cruzada con 3 particiones
- **scoring='roc_auc':** Métrica a optimizar
- **best_estimator_:** Modelo con los mejores hiperparámetros encontrados

1.12 Bloque 9: Evaluación Detallada del Mejor Modelo

1.12.1 ¿Qué se busca?

Realizar un análisis exhaustivo del rendimiento del modelo optimizado mediante múltiples visualizaciones, métricas y técnicas de validación.

1.12.2 ¿Por qué se aplica?

Una evaluación completa es esencial para:

- Entender fortalezas y debilidades del modelo
- Identificar tipos de errores (falsos positivos vs falsos negativos)
- Validar que el modelo generaliza bien (validación cruzada)
- Descubrir qué características son más importantes para las predicciones
- Comunicar resultados a stakeholders de manera clara

1.12.3 ¿Qué aporta al notebook?

9.1 Matriz de Confusión y Curvas:

- **Matriz de confusión:** Visualiza verdaderos/falsos positivos y negativos
- **Curva ROC:** Muestra trade-off entre tasa de verdaderos/falsos positivos
- **Curva Precision-Recall:** Especialmente útil para datos desbalanceados

9.2 Importancia de Características:

- **Top 10 features más importantes:** Identifica qué variables influyen más
- **Visualización de importancia:** Gráfico de barras ordenado
- **Insights de negocio:** Guía estrategias de retención basadas en factores clave

9.3 Validación Cruzada:

- **5-fold stratified CV:** Evalúa estabilidad del modelo
- **Distribución de scores:** Muestra consistencia entre folds
- **Media y desviación estándar:** Cuantifica variabilidad del rendimiento

1.12.4 Glosario contextualizado

- **Matriz de confusión:** Tabla 2x2 con TP, TN, FP, FN
 - **Verdaderos Positivos (TP):** Churn correctamente predicho
 - **Verdaderos Negativos (TN):** No churn correctamente predicho
 - **Falsos Positivos (FP):** Predicho churn pero no ocurrió (Tipo I)
 - **Falsos Negativos (FN):** No predicho churn pero sí ocurrió (Tipo II)
 - **Curva ROC:** Receiver Operating Characteristic curve
 - **TPR (True Positive Rate):** Recall, sensibilidad
 - **FPR (False Positive Rate):** Proporción de falsos positivos
 - **Curva Precision-Recall:** Muestra trade-off precision vs recall
 - **Feature importance:** Medida de cuánto contribuye cada variable
 - **StratifiedKFold:** Validación cruzada que mantiene proporción de clases
 - **cross_val_score:** Función que ejecuta validación cruzada
 - **Boxplot de CV scores:** Visualiza distribución de scores entre folds
-

1.13 Bloque 10: Conclusiones y Recomendaciones (Generadas Dinámicamente)

1.13.1 ¿Qué se busca?

Generar automáticamente un resumen ejecutivo con conclusiones, insights y recomendaciones basadas en los resultados reales obtenidos en la ejecución actual del notebook.

1.13.2 ¿Por qué se aplica?

Las conclusiones dinámicas son valiosas porque:

- Se actualizan automáticamente con cada ejecución
- Reflejan los resultados reales (especialmente importante en modo experimental)
- Proporcionan interpretación de negocio de las métricas técnicas
- Facilitan la comunicación con stakeholders no técnicos
- Documentan automáticamente cada experimento

1.13.3 ¿Qué aporta al notebook?

- **Resumen del mejor modelo:** Métricas de rendimiento interpretadas
- **Top 5 características más importantes:** Factores clave de churn
- **Comparación con baselines:** Cuantifica mejoras obtenidas
- **Impacto de SMOTE:** Evalúa efectividad del balanceo
- **Recomendaciones de negocio:**
 - Sistema de scoring de riesgo de churn
 - Estrategias de retención proactiva
 - Mejora de servicios basada en insights
 - Monitoreo continuo del modelo
- **Próximos pasos técnicos:** Deployment, API, dashboard, reentrenamiento

1.13.4 Glosario contextualizado

- **Conclusiones dinámicas:** Resultados generados automáticamente por código
 - **f-string:** Formato de cadenas en Python que inserta variables
 - **Interpretación de métricas:** Traducir números técnicos a significado de negocio
 - **ROC-AUC > 0.8:** Considerado “muy buena capacidad discriminativa”
 - **Recall bajo:** Indica que muchos casos de churn no son detectados
 - **Precision moderada:** Algunos clientes predichos como churn no lo harán
 - **Sistema de scoring:** Asignar probabilidad de churn a cada cliente
 - **Umbral de riesgo:** Punto de corte para decidir qué clientes contactar
 - **Retención proactiva:** Actuar antes de que el cliente decida irse
 - **Monitoreo de modelo:** Vigilar que el rendimiento no se degrade con el tiempo
-

1.14 Bloque 11: Guardar el Modelo y Verificar Tamaño para Deployment

1.14.1 ¿Qué se busca?

Serializar el modelo entrenado, el preprocessor y los metadatos en archivos para su posterior uso en producción, verificando que el tamaño sea adecuado para deployment en plataformas gratuitas.

1.14.2 ¿Por qué se aplica?

El guardado del modelo es esencial para:

- Usar el modelo entrenado sin tener que reentrenarlo
- Deployar el modelo en aplicaciones web o APIs
- Compartir el modelo con otros equipos
- Verificar que el tamaño sea compatible con plataformas de deployment gratuitas
- Mantener consistencia entre entrenamiento y predicción (mismo preprocessor)

1.14.3 ¿Qué aporta al notebook?

- **Detección automática de entorno:** Identifica si está en Colab o local
- **Guardado en Google Drive:** Persistencia permanente de archivos
- **Tres archivos guardados:**

1. **churn_model.pkl**: Modelo Random Forest (~43 MB)
 2. **preprocessor.pkl**: Pipeline de preprocesamiento (~0.01 MB)
 3. **metadata.json**: Información del modelo (métricas, fecha, parámetros)
- **Verificación de tamaño**: Evalúa compatibilidad con Render/Railway
 - **Estimación de RAM**: Calcula memoria necesaria en producción (~130 MB)
 - **Instrucciones de descarga**: Guía para obtener los archivos

1.14.4 Glosario contextualizado

- **Serialización**: Convertir objeto Python a archivo binario
 - **pickle**: Módulo de Python para serializar objetos
 - **joblib**: Alternativa a pickle optimizada para arrays grandes
 - **pkl file**: Archivo pickle serializado
 - **Preprocessor**: Pipeline que transforma datos crudos a formato del modelo
 - **Metadata**: Información sobre el modelo (versión, métricas, fecha)
 - **JSON**: Formato de texto para almacenar datos estructurados
 - **Google Drive mount**: Conectar Google Drive a Colab
 - **Deployment**: Poner el modelo en producción para uso real
 - **Render/Railway**: Plataformas de hosting gratuitas con límites de RAM
 - **RAM en producción**: Memoria necesaria para cargar y ejecutar el modelo
 - **os.path.getsize()**: Función que obtiene tamaño de archivo en bytes
-

1.15 Bloque 12: Resumen Técnico del Proyecto

1.15.1 ¿Qué se busca?

Proporcionar una documentación concisa de la metodología completa aplicada, tecnologías utilizadas y métricas de evaluación del proyecto.

1.15.2 ¿Por qué se aplica?

Un resumen técnico es importante para:

- Documentar el proceso completo de manera estructurada
- Facilitar la comprensión del proyecto a nuevos colaboradores
- Servir como referencia rápida de lo implementado
- Demostrar rigor metodológico
- Listar tecnologías para replicabilidad

1.15.3 ¿Qué aporta al notebook?

- **Checklist de metodología**: 10 pasos aplicados con
 - Análisis exploratorio completo
 - Limpieza de datos
 - Feature engineering (6 nuevas características)
 - Pipeline de preprocesamiento
 - 7 modelos evaluados
 - Manejo de desbalanceo con SMOTE

- Optimización de hiperparámetros
- Validación cruzada estratificada
- Métricas apropiadas para desbalanceo
- Análisis de interpretabilidad
- **Stack tecnológico:** Python, Pandas, NumPy, Scikit-learn, XGBoost, Imbalanced-learn, Matplotlib, Seaborn
- **Métricas utilizadas:** ROC-AUC, Precision, Recall, F1-Score, Matriz de Confusión
- **Información del dataset:** Telco Customer Churn de Kaggle

1.15.4 Glosario contextualizado

- **Metodología:** Conjunto de pasos sistemáticos aplicados al proyecto
 - **Stack tecnológico:** Conjunto de tecnologías y librerías utilizadas
 - **Pipeline robusto:** Secuencia de transformaciones bien estructurada
 - **Encoding:** Codificación de variables categóricas
 - **Scaling:** Normalización de variables numéricas
 - **Ensemble learning:** Combinación de múltiples modelos (Random Forest, Gradient Boosting)
 - **Gradient Boosting avanzado:** XGBoost, implementación optimizada
 - **Datos desbalanceados:** Dataset con clases de tamaños muy diferentes
 - **Media armónica:** Tipo de promedio usado en F1-Score
 - **Kaggle:** Plataforma de competencias y datasets de Data Science
-

1.16 Bloque 13: Generación de Informe Automático

1.16.1 ¿Qué se busca?

Crear automáticamente un informe completo en formato Markdown que documenta todos los resultados del análisis, guardándolo en Google Drive para fácil acceso y compartición.

1.16.2 ¿Por qué se aplica?

La generación automática de informes es valiosa porque:

- Documenta cada ejecución sin esfuerzo manual
- Crea reportes profesionales listos para compartir
- Mantiene consistencia en el formato
- Facilita comparación entre diferentes ejecuciones
- Permite compartir resultados con stakeholders no técnicos

1.16.3 ¿Qué aporta al notebook?

Genera un informe Markdown completo con 8 secciones:

1. **Resumen del dataset:** Dimensiones, variables, distribución de churn
2. **Calidad de datos:** Valores nulos, tipos de datos, limpieza realizada
3. **Resultados del mejor modelo:** Métricas detalladas, matriz de confusión
4. **Top 10 features importantes:** Ranking con porcentajes de importancia
5. **Parámetros del modelo optimizado:** Hiperparámetros encontrados

6. **Conclusiones y recomendaciones:** Insights de negocio
7. **Próximos pasos:** Checklist de acciones futuras
8. **Información técnica:** Metadata del notebook y ejecución

Archivo generado: Telco_Churn_Report_YYYY-MM-DD_HH-MM-SS.md en Google Drive

1.16.4 Glosario contextualizado

- **Markdown:** Lenguaje de marcado ligero para crear documentos formateados
 - **Informe automático:** Documento generado por código sin intervención manual
 - **Template de informe:** Estructura predefinida del documento
 - **Timestamp:** Marca de tiempo (fecha y hora) de la ejecución
 - **Formato tabular:** Presentación de datos en tablas
 - **Sección ejecutiva:** Resumen para audiencia no técnica
 - **Metadata del modelo:** Información sobre versión, fecha, parámetros
 - **Checklist de próximos pasos:** Lista de tareas pendientes
 - **strftime():** Función para formatear fechas como texto
 - **write():** Método para escribir contenido en archivo
 - **Persistencia de resultados:** Guardar resultados para acceso futuro
-

1.17 Conclusión General del Análisis

Este notebook representa un **proyecto completo y profesional de Machine Learning** que sigue las mejores prácticas de la industria:

1.17.1 Fortalezas del Proyecto

1. **Metodología rigurosa:** Desde EDA hasta deployment
2. **Reproducibilidad:** Sistema de semillas aleatorias configurable
3. **Manejo profesional de datos:** Detección y corrección de anomalías
4. **Feature engineering inteligente:** 6 características derivadas con sentido de negocio
5. **Evaluación exhaustiva:** 7 modelos comparados con múltiples métricas
6. **Manejo de desbalanceo:** SMOTE aplicado correctamente
7. **Optimización eficiente:** RandomizedSearchCV balanceando tiempo y precisión
8. **Validación robusta:** Validación cruzada estratificada
9. **Interpretabilidad:** Análisis de feature importance
10. **Documentación automática:** Informes generados dinámicamente
11. **Preparado para producción:** Modelo guardado con verificación de tamaño

1.17.2 Aplicabilidad en el Contexto de Telecomunicaciones

El modelo desarrollado permite a la empresa:

- **Identificar clientes en riesgo** con ~83% de precisión (ROC-AUC)
- **Priorizar acciones de retención** basadas en probabilidad de churn
- **Entender factores clave** que influyen en la decisión de abandonar
- **Optimizar recursos** enfocándose en clientes de alto riesgo
- **Medir impacto** de estrategias de retención

1.17.3 Recomendaciones para Mejora Continua

1. **Reentrenamiento periódico:** Actualizar modelo con datos recientes
2. **A/B testing:** Validar efectividad de estrategias de retención
3. **Monitoreo de drift:** Detectar cambios en patrones de datos
4. **Expansión de features:** Incorporar datos de uso, quejas, competencia
5. **Modelos más avanzados:** Probar redes neuronales, stacking, etc.

Documento generado como parte de la asesoría técnica del proyecto Telco Customer Churn Prediction