|  |  |  |
| --- | --- | --- |
| **SECCIÓN** | **BIBLIOTECAS** | **ACCIONES** |
| LECTURA DE LOS DATOS | * import pandas as pd | * Se leen los datos desde la planilla CSV resultante del primer desafío TelecomX, mostrando las 28 columnas que la conforman y sus tipos de datos, y que son 7267 registros. |
| EXPLORANDO LOS DATOS |  |  |
| VARIABLES CATEGÓRICAS | * import seaborn as sns * import plotly.express as px | * Se muestran gráficos de las variables categóricas. |
| VARIABLES NUMÉRICAS |  | * Se muestran gráficos de las variables numéricas. |
| CORRELACIONES | * import seaborn as sns * import matplotlib.pyplot as plt * import warnings | Se muestran gráficos con mapas de calor:   * Churn con Perfil de cliente. * Churn con servicios contratados (resumen). * Churn con servicios contratados (detalle). * Churn con datos contractuales. |
| TRANSFORMACIÓN DE LOS DATOS |  |  |
| VARIABLES EXPLICATIVAS Y VARIABLE DE RESPUESTA |  | * Se deja en x todas las variables explicativas (dejando fuera solo Churn). * Se deja en y la variable de respuesta Churn. |
| TRANSFORMANDO LAS VARIABLES EXPLICATIVAS | * from sklearn.compose import make\_column\_transformer * from sklearn.preprocessing import OneHotEncoder | * Se transforman las variables explicativas mediante OneHotEncoder, dejando solo variables con valores 0 y 1 para las categóricas, y manteniendo sin cambios las numéricas. |
| TRANSFORMANDO LA VARIABLE RESPUESTA | * from sklearn.preprocessing import LabelEncoder | * Se transforma la variable de respuesta (Churn) mediante LabelEncoder, dejando solo valores 0 y 1. |
| AJUSTANDO MODELOS |  |  |
| DIVIDIENDO LOS DATOS (ENTRENAMIENTO Y PRUEBA) | * from sklearn.model\_selection * import train\_test\_split | * Se dividen los datos en entrenamiento (train) y prueba (test). |
| MODELO DE REFERENCIA – BASELINE | * from sklearn.dummy import DummyClassifier * from sklearn.dummy import DummyRegressor * from sklearn.model\_selection import train\_test\_split * from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score | * Se genera el modelo de referencia (dummy) resultando un score de 0,712. * Se generan métricas RMSE (0,447), MAE (0,407) y R2 (-0,0018). |
| ÁRBOLES DE DECISIÓN | * from sklearn.tree import DecisionTreeClassifier * from sklearn.tree import plot\_tree * import matplotlib.pyplot as plt * from sklearn.ensemble import RandomForestRegressor * from yellowbrick.regressor import prediction\_error * from yellowbrick.regressor import residuals\_plot | * Se genera el modelo de árbol resultando un score de 0,772. * Se grafica el resultado del modelo de árbol (profundidad 4) * Se generan métricas RMSE (0,382), MAE (0,294) y R2 (0,267). * Se muestran gráficos de errores de predicción y de residuales. |
| NORMALIZANDO LOS DATOS | * from sklearn.preprocessing import MinMaxScaler | * Se normalizan los datos mediante MinMaxScaler. |
| KNN | * from sklearn.neighbors import KNeighborsClassifier | * Se genera el modelo KNN resultando un score de 0,742. |
| ESCOGIENDO Y SERIALIZANDO EL MEJOR MODELO | * from sklearn.model\_selection import KFold, cross\_validate * import pickle | * Se escoge y serializa el modelo de árbol. |
| PREDICIENDO VARIABLE RESPUESTA PARA NUEVO DATO |  | * Se define un nuevo dato y se predice la variable respuesta con modelo de árbol, |
|  |  |  |
| SELECCIÓN DE VARIABLES EXPLICATIVAS MÁS RELEVANTES | * from sklearn.model\_selection import GridSearchCV | * Se determina la importancia de cada variable explicativa, se ordenan de mayor a menor y seleccionan las primeras cinco, cuya importancia es mayor a 4,5. * Se define una grid de combinaciones de parámetros para el modelo de árbol (max\_depth, min\_samples\_leaf, min\_samples\_split y n\_estimators), con tres alternativas para cada uno, resultando los mejores parámetros con las siguientes métricas RMSE (0,388), MAE (0,299) y R2 (0,253). * Las cuatro variables más relevantes, con importancia mayor al 4%, son las siguientes: Contract\_Month-to-month (55.15%), InternetService\_Fiber optic (17.41%), tenure (17.20%) y Charges.Total (4.64%). |
| MODELO ÁRBOL (CON VARIABLES EXPLICATIVAS MÁS RELEVANTES) | * from sklearn.tree import DecisionTreeClassifier | * El modelo de árbol con variables explicativas más relevantes resulta con un score de 0,780. |
| VALIDANDO Y EVALUANDO EL MODELO | * from sklearn.model\_selection import train\_test\_split * from sklearn.metrics import confusion\_matrix * from sklearn.metrics import ConfusionMatrixDisplay * from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score | * Se valida y evalúa el modelo de árbol, resultando un score de 0,985 para el conjunto de entrenamiento y de 0,714 para el conjunto de validación. * Se genera y muestra gráfico de la matriz de confusión. * Resultando una precisión de 0,505, una sensibilidad de 0,476 y una exactitud de 0,714. Además resulta un F1-score de 0,490. |
| MODELO KNN (CON VARIABLES EXPLICATIVAS MÁS RELEVANTES) | * from sklearn.preprocessing import MinMaxScaler * from sklearn.neighbors import KNeighborsClassifier | * El modelo KNN con variables explicativas más relevantes resulta con un score de 0,753. |
|  |  |  |
| VALIDACIÓN CRUZADA | * from sklearn.model\_selection import KFold, cross\_validate | * Resulta un promedio de exactitud igual a 0,744 con un desvío típico de 0,000923, y un intervalo de confianza entre 0,728 y 0,765. |
| VALIDACIÓN CRUZADA CON SENSIBILIDAD |  | * Resulta un intervalo de confianza entre 0,446 y 0,597. |
| ESTRATIFICANDO LOS DATOS | * from sklearn.model\_selection * import StratifiedKFold | * Churn igual a Yes se presenta en el 28,8% de los casos. * Resulta un intervalo de confianza entre 0,461 y 0,553. |
| BALANCEO DE DATOS |  |  |
| OVERSAMPLING | * from imblearn.over\_sampling import SMOTE * import pandas as pd | * Resulta un intervalo de confianza entre 0,754 y 0,828. |
| PIPELINE PARA VALIDACIÓN | * from imblearn.pipeline import Pipeline as imbpipeline | * Resulta un intervalo de confianza entre 0,523 y 0,638. |
| UNDERSAMPLING | * from imblearn.under\_sampling * import NearMiss | * Resulta un intervalo de confianza entre 0,611 y 0,707. |
| PROBANDO EL MODELO | * from sklearn.metrics import classification\_report | * Se genera y grafica una matriz confusión. |

A continuación se incluyen algunos gráficos relevantes.

CANTIDAD DE ABANDONOS DE CONTRATO (CHURN) VS TIPO DE CONTRATO:

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

La mayor parte de los abandonos se da para los contratos Month-to\_Month

CANTIDAD DE ABANDONOS DE CONTRATO (CHURN) VS MÉTODO DE PAGO

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

La mayor parte de los abandonos se da para los métodos de pago Electronic check

DISTRIBUCIÓN (BOXPLOT) DE MESES DE CONTRATO (TENURE) vs ABANDONOS DE CONTRATO (CHURN)

Gráfico, Gráfico de cajas y bigotes

El contenido generado por IA puede ser incorrecto.

La mayor parte de los abandonos se da al inicio del contrato