Reporte técnico Proyecto Churns Telco

Luciano De Doménico - Francisco Lavaggi

Noviembre 2024

1 Introducción y objetivos

El presente reporte amplía la información técnica acerca del modelo de Machine Learning entrenado y utilizado para predecir la deserción de clientes de la empresa Telco NN a partir de un conjunto de datos acerca de los usuarios. El objetivo principal radica en generar una herramienta predictora capaz de identificar grupos de clientes con amplias probabilidades de anular su contrato con el proveedor de servicio de telecomunicaciones en el corto plazo, con el menor error posible. Esta información permitiría a la empresa anticiparse y emplear acciones para evitar las deserciones.

Se generó una máquina estadística capaz de predecir las deserciones con un 80% de precisión. Para entrenarla se utilizaron datos históricos de los usuarios de la empresa, tanto sobre aquellos quienes han solicitado prescindir de los servicios de Telco NN en el pasado, como de quienes aún cuentan con un contrato activo.

Se entrenaron dos modelos de aprendizaje supervisado con dos diferentes algoritmos (Support Vector Machine y KNN), se midió la performance de cada uno de ellos (con un set de datos preservados para la evaluación), se los comparó entre sí y se seleccionó el modelo más apropiado.

También se aplicó una técnica de reducción de dimensionalidad (PCA) para buscar simplificar el problema y obtener mejores resultados luego de re-entrenar al modelo.

Finalmente se concluyó acerca del mejor modelo hallado para resolver este problema y se indicaron accionables para seguir potenciando la herramienta en un futuro.

2 Descripción del dataset

El dataset provisto por la empresa contiene originalmente 7043 muestras y 22 parámetros (o features) asociados a cada muestra. Algunos de los features más relevantes son:

- Gender - *género del cliente*
- InternetService - *indica si el cliente tiene contratado el servicio de internet*
- MonthlyCharges - *costo mensual para el cliente*
- PhoneService - *indica si el cliente tiene contratado el servicio de telefonía*
- Contract - *indica el tipo de contrato del cliente (mensual, anual, bianual)*
- Churn - **variable a predecir** - *indica si el cliente dejó la compañía o no*

3 Análisis exploratorio de datos

Previo a entrenar un modelo de Machine Learning, resulta mandatorio efectuar el pre-procesamiento del dataset para evitar ingresar "ruido" al modelo. Se realizaron los siguientes ajustes:

- Remoción de features innecesarios para la predicción. - *Unnamed y CustomerID*
- Pre-análisis estadístico sobre features numéricos - *con el fin de identificar posibles inconsistencias*
- Identificación y reemplazo de valores nulos - *reemplazados con "moda" para valores categóricos nulos y "media" para valores numéricos nulos*
- Búsqueda y corrección de errores de escritura en features categóricos
- Pre-análisis de distribuciones de probabilidad
- Búsqueda de Outliers - *No se hallaron outliers*
- Análisis de correlación entre cada par de features - *con el fin de identificar variables linealmente correlacionadas y disminuir la dimensionalidad del dataset. Ver figura 1*

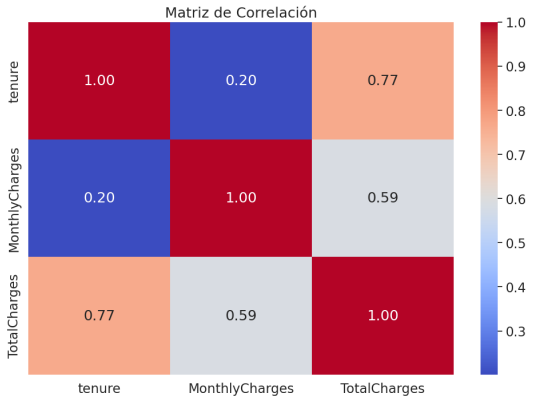

Figure 1: Matriz de correlación entre las variables numéricas del dataset

4 Materiales y métodos

El problema en estudio debe resolverse con un modelo de aprendizaje supervisado de clasificación (clasificador) con 2 clases. El clasificador debe predecir la clase perteneciente de sus futuros inputs, las cuales estarán asociados a los dos escenarios posibles (*El cliente prescindirá o no de los servicios de la empresa.*).

El algoritmo clasificador elegido en primer lugar fue Support Vector Machine, el cual buscará discriminar cada muestra con un hiperplano separador en un entorno multi-dimensional.[2]

Se aplicó también KNN como algoritmo alternativo con la finalidad de contrastar los rendimientos de ambos y utilizar el que mejor logre predecir las deserciones de Telco NN.

Antes de entrenar ambos modelos, se deben hacer algunas últimas transformaciones en los datos para optimizar las performances de los modelo y facilitar el trabajo de los algoritmos.

En primer lugar se ajustaron todas las variables categóricas. La técnica utilizada consiste en transformar los valores en 0 ó 1 a partir de la creación de features binarios auxiliares (llamados "dummies") que reemplazarán a los originales.

A modo de ejemplo, la variable "Gender", que originalmente contenía a los valores "Female" o "Male". se transformó en una variable llamada "Gender_Male" que valdrá "1" si el usuario de la muestra es hombre y "0" si se trata de una mujer. Dicho reemplazo se aplica para cada una de las variables catégoricas. Se debe considerar que para reemplazar variables categóricas que contengan más de dos textos posibles, se necesita más de una dummie. Esto produjo que nuestro dataset amplíe su cantidad de columnas, siendo originalmente 22 y resultando en

31 luego de la transformación.

En segundo lugar se hizo una partición de las muestras totales en dos grupos: El primer grupo de muestras (x-train) será utilizado para entrenar al modelo y el segundo grupo (x-test) será utilizado para evaluar el modelo. El grupo de entrenamiento incluye el 80% de las muestras, mientras que el grupo de evaluación contiene al 20% restante de los datos.

En tercer lugar se escalaron los valores de las features para estandarizar las distribuciones de valores que cada columna contiene. Aplicamos "Standarization"[1]. En la figura 2 se visualiza la distribución de los rangos originales de cada variable numérica del dataset.

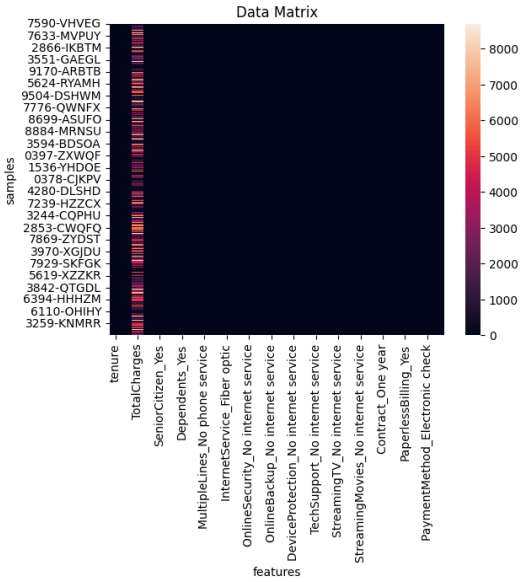

Figure 2: Rangos originales de valores en features

Standarization transforma cada valor del dataset para llevarlos a un rango con media $\mu = 0$ y desvío estándar $\sigma = 1$. Se obtiene una distribución normal estándar. En la figura 3 se visualizan los rangos de cada variable posteriores a aplicar standarization.

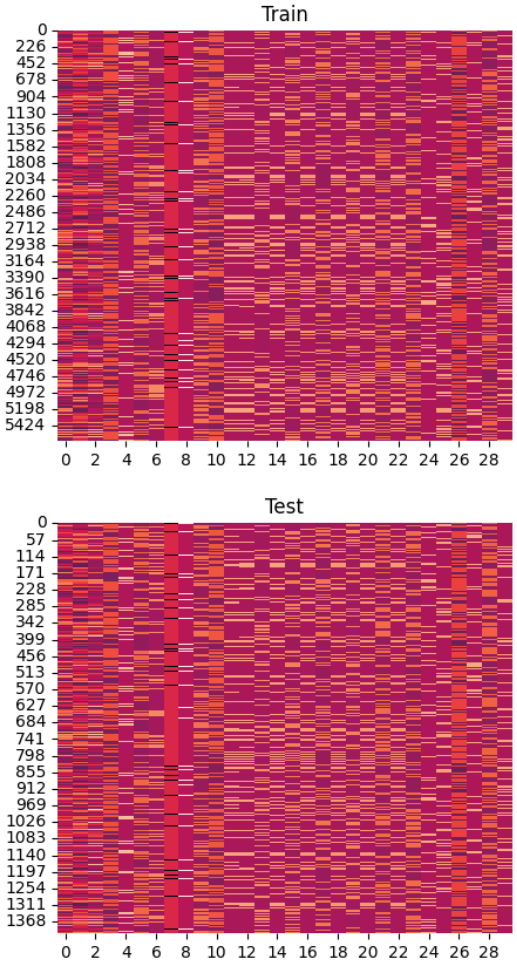

Figure 3: Distribución en test y train luego de aplicar standarization

5 Experimentos y resultados

5.1 Support Vector Machine

Se realizaron múltiples iteraciones de entrenamiento con SVM asociados a diferentes valores de hiperparámetros. Se concluyó que los mejores hiperparámetros para este modelo son equivalentes a C = 10, gamma = 0.001 y kernel = 'rbf'. Bajo estas condiciones se alcanzó un accuracy de 81,76% con el set de test. En la figura 4 se traza la curva ROC para evaluar los resultados de este modelo.

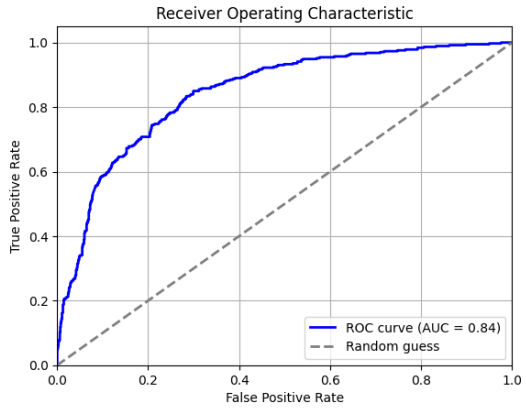

Figure 4: Curva ROC - SVM

En la figura 5 se observa la matriz de confusión del modelo, generada a partir de los resultados de las predicciones en test comparadas con su respectiva "Ground Truth".

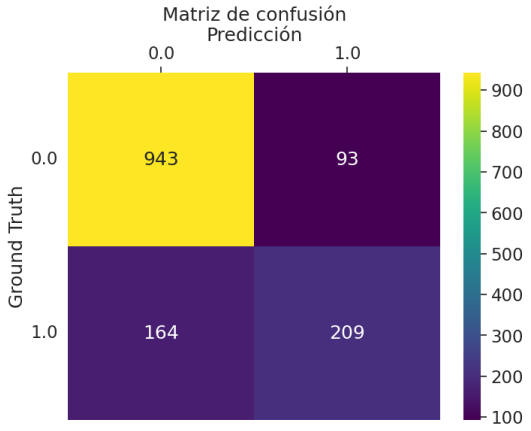

Figure 5: Matriz de confusión - SVM

5.2 K-Nearest Neighbors

Adicionalmente se replicaron las pruebas con otro algoritmo de clasificación: KNN [3]. Se compararon los rendimientos entre ambos modelos para discernir cuál resulta mejor para resolver este problema en específico. El accuracy de KNN fue de 77,78%, siendo ligeramente menor al obtenido con SVM. Por lo tanto, **se definió avanzar utilizando SVM.**

5.3 Principal Component Analysis

Es posible proyectar el conjunto de datos original en un subespacio simplificado, con menor cantidad de dimensiones (componentes principales). Este procedimiento se denomina Principal Component Analysis (PCA) y busca perder la menor cantidad de información posible en el proceso. Lo aplicamos para este problema y analizaremos como responde el modelo ante un input reducido.

PCA utiliza las distancias entre los valores de los datos para generar las proyecciones, es decir que no puede ser aplicado sobre conjuntos de datos que contengan "dummies" de variables categóricas. Por este motivo se empleó la técnica sobre un conjunto de datos con las columnas categóricas excluidas, resultando en una matriz de únicamente tres columnas. Dicha matriz reducida de todas formas será transformada con PCA para obtener matemáticamente sus tres componentes principales y evaluar si dicho procedimiento genera algún efecto positivo al utilizarse para entrenar el modelo de SVM. En la figura 6 se puede ver la distribución de las muestras proyectadas en las dos componentes principales. Con las 2 PC se explica el 96% del dataset sin dummies.

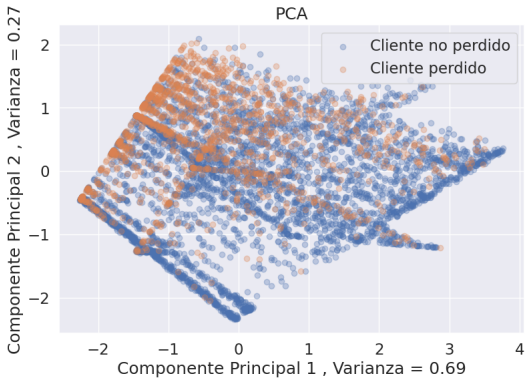

Figure 6: Proyección en 2 PC - *Nota: Observamos cierta linealidad en la nube de puntos, la atribuímos a que MonthlyCharges y TotalCharges están algo correlacionadas (0,59 de coef de Pearson)*

Luego del entrenamiento del modelo con el nuevo conjunto de datos transformado se obtuvo un accuracy del 78,7% en test.

6 Discusión y conclusiones

Se concluyó que el mejor modelo hallado en la etapa de experimentación es SVM sin PCA aplicado. Atribuimos este hecho a la naturaleza del dataset original, que contiene una baja relación de features numéricos sobre features totales. Gran parte del dataset contiene datos categóricos y binarios, lo que no favorece a la técnica de PCA para arrojar resultados convenientes.

El mejor modelo hallado tiene un accuracy de alrededor del 80%, lo que puede considerarse en primera instancia como un resultado aceptable. Para conseguir aún mejores resultados se sugiere a la empresa continuar recolectando datos para re-entrenar al modelo en el futuro.

Referencias y Bibliografía

[1] StandardScaler: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[2] Mariette Awad, Rahul Khanna. Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers (Abril 2015).

[3] Pádraig Cunningham, Sarah Jane Delany. k-Nearest Neighbour Classifiers (Abril 2015): https://arxiv.org/pdf/2004.04523

[4] Gareth James, Daniela Witten, Trevor Hastie, Rob Tibshirani, Jonathan Taylor. An Introduction to Statistical Learning with applications in Python: https://www.statlearning.com/

[5] Documentación de Numpy: https://numpy.org/doc/stable/

[6] Documentación de Pandas: https://pandas.pydata.org/docs/user_guide.html

[7] Documentación de Scikitlearn: https://scikit-learn.org/stable/user_guide.html