Documentación Técnica: Sistema Predictivo de Retención de Clientes (Churn)

1. Resumen del Proyecto

Implementación de una solución analítica avanzada que integra Machine Learning y Business Intelligence para predecir la probabilidad de abandono de clientes en una empresa de telecomunicaciones. El objetivo es transformar datos históricos en estrategias preventivas de fidelización.

2. Arquitectura de Datos y ML

- **Origen de Datos:** Dataset de telecomunicaciones con perfiles de usuario, servicios contratados y estado de permanencia (Churn).
- **Procesamiento (ETL):** Realizado en Python utilizando la librería Pandas para la limpieza de datos, manejo de valores nulos y normalización.
- **Ingeniería de Características:** Aplicación de Label Encoding para convertir variables categóricas (como tipo de contrato y servicio de internet) en formatos procesables por el modelo.
- **Modelado Predictivo:** Uso del algoritmo Random Forest Classifier de la librería Scikit-learn, logrando una precisión de entrenamiento del 79%.

3. Implementación Analítica (DAX)

Se desarrollaron medidas personalizadas en Power BI para validar el rendimiento del modelo contra la realidad:

- **Tasa de Fuga Real:** Calculada mediante DIVIDE y CALCULATE sobre la columna original de Churn.
- **Tasa de Fuga Predicha:** Basada en los resultados generados por el modelo de Machine Learning.
- **Exactitud del Modelo:** Comparación lógica fila por fila entre la predicción y el dato real para monitoreo de drift del modelo.

4. Visualización y KPIs Estratégicos

El dashboard fue diseñado siguiendo principios de UX/UI ejecutivo, destacando tres pilares:

- **% Fuga Real:** Diagnóstico del estado actual del negocio.
- **Precisión de IA:** Nivel de confianza del modelo predictivo (79%).
- **Alertas de Fuga:** Cantidad total de clientes identificados por la IA en riesgo de abandono.

5. Desarrollo en Python y Validación del Modelo

Utilicé Visual Studio Code como entorno de desarrollo principal. El proceso incluyó la limpieza de datos con Pandas y la creación de visualizaciones técnicas para entender el comportamiento de las variables antes de su integración en el dashboard final.

5.1. Entrenamiento del Modelo - Random Forest

Implementé el algoritmo Random Forest Classifier, seleccionado por su robustez ante datos ruidosos y su capacidad para manejar relaciones no lineales. El modelo fue entrenado utilizando un conjunto de datos procesado mediante técnicas de Label Encoding.

```python
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score

# 1. Definir X (lo que usamos para predecir) y (lo que queremos adivinar: Churn)
X = df_ml.drop('Churn', axis=1)
y = df_ml['Churn']

# 2. Dividir datos: 80% para entrenar y 20% para la evaluación final
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 3. Creación y entrenamiento del modelo
modelo = RandomForestClassifier(n_estimators=100, random_state=42)
modelo.fit(X_train, y_train)

# 4. Hacer predicciones con el 20% que el modelo no conoce
predicciones = modelo.predict(X_test)

# 5. Ver qué tan bueno es nuestro sistema
print(f"Precisión del modelo: {accuracy_score(y_test, predicciones):.2f}")
print("\nReporte de Clasificación:")
print(classification_report(y_test, predicciones))
```

Python

Figura 1: Implementación del entrenamiento del modelo RandomForestClassifier mediante Scikit-Learn.

5.2. Evaluación de Métricas de Desempeño

Tras el entrenamiento, el modelo fue evaluado con un set de datos de prueba independiente. Los resultados arrojaron una exactitud (Accuracy) del 79%, lo que representa un alto nivel de confiabilidad para la toma de decisiones comerciales.

```
Precisión del modelo: 0.79

Reporte de Clasificación:
              precision    recall  f1-score   support

           0       0.83      0.91      0.86      1033
           1       0.64      0.47      0.55       374

    accuracy                           0.79      1407
   macro avg       0.73      0.69      0.70      1407
weighted avg       0.78      0.79      0.78      1407
```

Figura 2: Reporte de clasificación detallando la precisión, recall y F1-score del modelo predictivo.

5.3. Análisis de Importancia de Características (Feature Importance)

Como parte del proceso de interpretabilidad del modelo, generé una gráfica de importancia de variables. Este análisis técnico me permitió identificar que el tipo de servicio de internet y los cargos mensuales son los factores con mayor peso en la decisión de fuga de los clientes.

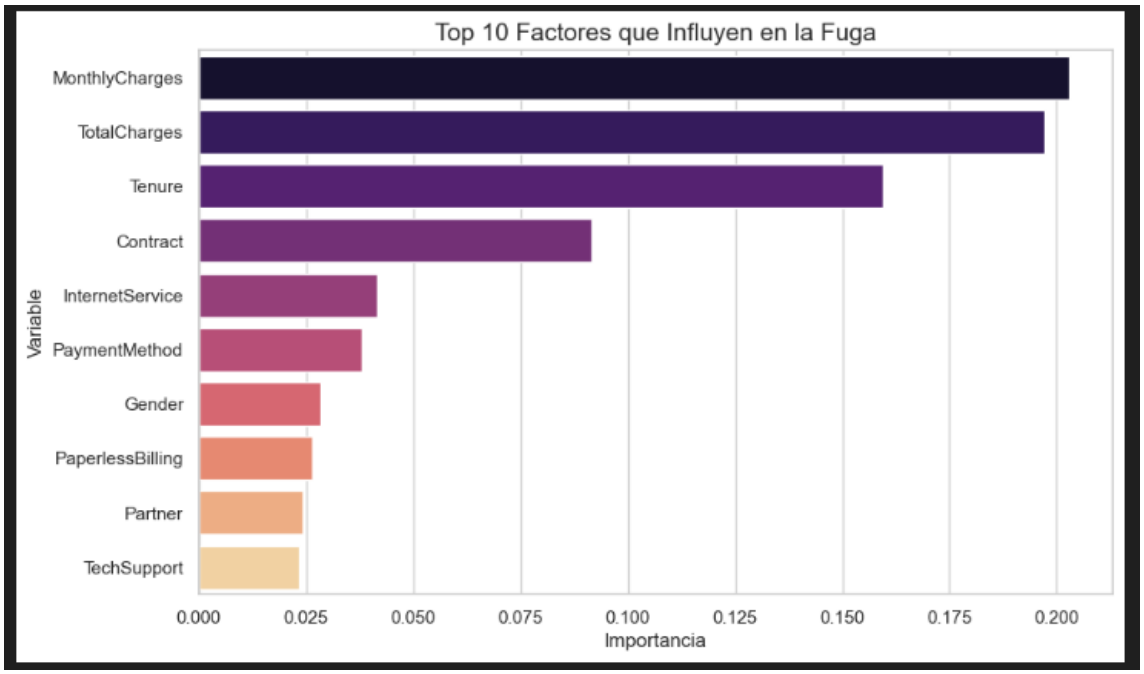

Figura 3: Visualización de las variables con mayor impacto predictivo dentro del algoritmo.

6. Visualización de Resultados – Dashboard Ejecutivo

Diseñé una interfaz interactiva en Power BI enfocada en la toma de decisiones basada en datos predictivos. El diseño prioriza la claridad de las métricas sobre la complejidad gráfica, facilitando la identificación inmediata de riesgos.

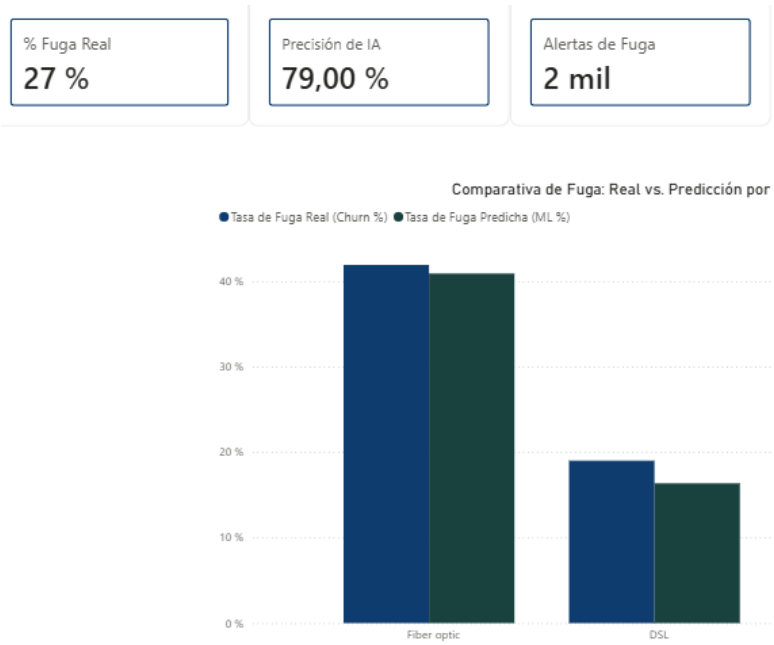

Figura 7: Interfaz final del Dashboard de Predicción de Churn, integrando métricas reales y resultados del modelo de Machine Learning.

7. Conclusiones

- **Impacto del Modelo:** La implementación del algoritmo Random Forest permitió identificar patrones de fuga con un 79% de exactitud, proporcionando una herramienta confiable para la retención proactiva de clientes.
- **Insights del Negocio:** Gracias al análisis de importancia de características, se determinó que el tipo de Servicio de Internet (Fibra Óptica) y los cargos mensuales son los principales detonantes del Churn, permitiendo dirigir las campañas de fidelización con mayor precisión.
- **Valor de la Integración:** El éxito del proyecto radica en la integración "End-to-End", demostrando que la combinación de Data Science (Python) y Business Intelligence (Power BI) optimiza la transformación de datos crudos en decisiones estratégicas.