

Actividad 1 – Machine Learning II

Profesor: Francisco Pérez Galarce

Formato de entrega. Un notebook en Python con: Código limpio, celdas ordenadas y comentarios. Gráficos y tablas de resultados.

Dentro del mismo notebook usando Markdown debe incluir: Descripción del preprocesamiento. Comparación de modelos (sin polinomios vs. con polinomios; sin penalización vs. penalizados). Discusión de resultados y conclusiones. La actividad puede ser desarrollada en grupos de máximo 3 personas.

El avance de cada grupo será discutido al final de la clase del próximo sábado 06 de diciembre.

CONTEXTO

Una empresa de telecomunicaciones desea predecir qué clientes tienen mayor probabilidad de fugarse (churn), para diseñar campañas de retención. Se cuenta con un dataset de clientes, sus características de uso y una etiqueta binaria que indica si el cliente se fue (churn = 1) o se mantuvo (churn = 0). El objetivo de esta actividad es construir y evaluar modelos de regresión logística, comparando:

1. Un modelo con *features* “básicas” (preprocesadas).
2. Un modelo con transformaciones polinomiales sobre las *features* numéricas.
3. Modelos con penalización (regularización) para controlar complejidad.

La evaluación debe realizarse mediante validación cruzada *k-fold*, utilizando matriz de confusión, curva ROC y curva *Precision-Recall*, en línea con lo visto en la clase.

Al finalizar la actividad, la o el estudiante será capaz de:

- Implementar un flujo completo de preprocesamiento de datos para un problema de churn.
- Entrenar modelos de regresión logística con y sin transformaciones polinomiales.
- Incorporar penalizaciones (L2, L1 o elastic net) para controlar sobreajuste.
- Evaluar clasificadores con *k-fold cross-validation* mediante:
 - ✓ Matriz de confusión y métricas asociadas (*accuracy*, *precision*, *recall*, *F1*).
 - ✓ Curva ROC y AUC.
 - ✓ Curva *Precision-Recall* y *PR-AUC*, considerando el desbalance del churn.

INSTRUCCIONES

Paso 0. Dataset y descripción de variables

1. Descargar y cargar en Jupyter Notebook el data set disponible en Blackboard.
2. Identificar:
 - ✓ Variable objetivo: churn (binaria: 1 = se va, 0 = se queda).
 - ✓ Variables numéricas (ej.: minutos de uso, cargos mensuales, duración del contrato...).
 - ✓ Variables categóricas (ej.: tipo de plan, método de pago, tipo de contrato, etc.).

Paso 1. Exploración y preprocesamiento de datos

En el notebook:

1. Exploración inicial
 - ✓ Mostrar tamaño del dataset.
 - ✓ Ver proporción de churn vs no churn (ver si hay desbalance de clases).
 - ✓ Describir estadísticos básicos de variables numéricas y conteos de variables categóricas.
2. Tratamiento de datos faltantes
 - ✓ Identificar valores faltantes.
 - ✓ Definir una estrategia: imputación simple (media / mediana para numéricas, categoría más frecuente para categóricas) o eliminar filas si es razonable.

3. Codificación de variables categóricas
 - ✓ Usar one-hot encoding / `get_dummies` o `OneHotEncoder` de `scikit-learn`.
 - ✓ Evitar multicolinealidad perfecta (por ejemplo, usando `drop_first=True`).
4. Escalamiento de variables numéricas
 - ✓ Normalizar o estandarizar las variables numéricas (por ejemplo, *StandardScaler*).
5. Definir matriz de features y vector objetivo
 - ✓ X: todas las variables explicativas preprocesadas.
 - ✓ y: variable binaria de churn.

Justificar brevemente las decisiones de preprocesamiento.

Paso 2. Modelo base: regresión logística con features obtenidas

1. Dividir el dataset en entrenamiento / evaluación usando validación cruzada *k-fold* estratificada (por ejemplo, *StratifiedKFold* con $k = 5$ o $k = 10$).
2. Entrenar un modelo de regresión logística sin transformaciones polinomiales, solo con las *features* preprocesadas:
 - ✓ Usar una configuración básica (por ejemplo, `penalty='none'`).
 - ✓ Asegurar convergencia (ajustar `max_iter` si es necesario).
3. Para cada *fold* de la validación cruzada:
 - ✓ Ajustar el modelo en el set de entrenamiento del fold.
 - ✓ Obtener predicciones de probabilidad y de clase en el set de validación.
 - ✓ Calcular: Matriz de confusión. Accuracy, precision, recall, F1. Curva ROC y AUC-ROC. Curva Precision-Recall y PR-AUC.
4. Resumir los resultados:
 - ✓ Tabla con promedios y desviaciones estándar de las métricas por *fold*.
 - ✓ Gráfico de ROC promedio y PR promedio.

Paso 3. Modelo con transformaciones polinomiales

1. Seleccionar un subconjunto de variables numéricas “relevantes” (por ejemplo, 3–5 variables clave) para generar términos polinomiales de grado 2:
 - ✓ Usar `PolynomialFeatures` (grado 2) para estas variables, incluyendo términos de interacción.
 - ✓ Combinar estas nuevas *features* con el resto de *features* (numéricas y categóricas codificadas).
2. Repetir el mismo esquema de *k-fold cross-validation*, pero ahora con este conjunto extendido de *features*:
 - ✓ Entrenar regresión logística sin penalización fuerte (o con una penalización mínima).
 - ✓ Evaluar con las mismas métricas y curvas (matriz de confusión, ROC, PR).
3. Comparar con el modelo base:
 - ✓ ¿Mejoran las métricas?
 - ✓ ¿Se observa indicio de sobreajuste? (por ejemplo, rendimiento muy bueno en entrenamiento pero no tanto en validación, coeficientes muy grandes, etc.).

Paso 4. Aplicar penalizaciones (regularización)

1. Considerar al menos dos variantes:
 - ✓ Regresión logística con penalización L2 (Ridge).
 - ✓ Regresión logística con penalización L1 (Lasso) o Elastic Net.
2. Para cada tipo de penalización:
 - ✓ Definir una grilla de valores de C (inversa de la fuerza de regularización).
 - ✓ Usar *grid search cross-validation* para seleccionar el mejor hiperparámetro según una métrica primaria (por ejemplo, F1 de la clase churn o PR-AUC).
3. Evaluar los modelos penalizados:
 - ✓ Calcular las mismas métricas y curvas que antes.
 - ✓ Comparar: Modelo sin polinomios vs con polinomios. Con y sin penalización.
 - ✓ Cómo cambian los coeficientes al aplicar regularización.

- ✓ Si la regularización ayuda a reducir sobreajuste que podría introducir el aumento de dimensionalidad por los polinomios.

Paso 5. Análisis crítico

1. ¿Qué tan desbalanceado está el problema de churn? ¿Por qué accuracy, por sí sola, puede ser engañosa en este contexto? Relacionar con la matriz de confusión.
2. Comparando los distintos modelos, ¿cuál seleccionarían para este problema? Justificar usando:
 - ✓ F1.
 - ✓ AUC-ROC.
 - ✓ PR-AUC.
3. ¿Qué efecto tuvieron las transformaciones polinomiales sobre el rendimiento y la complejidad del modelo?
4. ¿Cómo influyó la regularización en:
 - ✓ El rendimiento en validación.
 - ✓ La magnitud de los coeficientes.
 - ✓ La estabilidad de las métricas a través de los *folds*?
5. ¿Qué *trade-offs* observan entre:
 - ✓ Maximizar recall de churn (reducir falsos negativos).
 - ✓ Mantener una *precision* razonable (no saturar al equipo de retención con demasiados falsos positivos)? Relacionar con la curva *Precision–Recall* y el objetivo de negocio.
6. ¿Qué otras técnicas podría aplicar para mejorar el rendimiento en este conjunto de datos?