Examen: Evaluación de Modelos de Machine Learning y Redes Neuronales (Versión Estudiante)

Fecha de creación: 16/09/2025

Descripción

Este examen práctico evalúa sus habilidades para trabajar con modelos clásicos (Random Forest) y redes neuronales, utilizando *pipelines* de scikit-learn y aplicando una limpieza exhaustiva de datos. El dataset provisto es deliberadamente sucio (nulos, duplicados, outliers, formatos y categorías inconsistentes). Se requiere: (1) limpieza de datos, (2) modelado con Random Forest mediante pipeline, (3) diseño/entrenamiento de una red neuronal MLP, (4) comparación de resultados y (5) guardar y reutilizar el pipeline/modelo con datos de prueba.

Documentación del Dataset

Columnas: ID, Edad, Genero, Ingresos_Anuales, HistorialCredito, Casado, Default, FechaNacimiento, Telefono, Pais, CodigoPostal. Problemas: valores faltantes, duplicados, outliers, categorías/formatos heterogéneos, consistencia Pais↔CodigoPostal, normalización de FechaNacimiento y Telefono.

Parte 1: Exploración y Limpieza de Datos

1) Explorar tamaño, tipos y valores únicos. 2) Manejar faltantes (imputación/eliminación justificada). 3) Detectar y tratar outliers (percentiles/IQR). 4) Remover duplicados. 5) Estandarizar categorías en *Genero* y *Casado*. 6) Validar consistencia *Pais–CodigoPostal*. 7) Normalizar *FechaNacimiento* a YYYY-MM-DD y limpiar *Telefono*. 8) Documentar cada transformación y su justificación.

Parte 2: Pipeline con Random Forest (incluya SUS funciones de limpieza)

Construya un *pipeline* que encapsule sus propias funciones de limpieza/validación (usando *FunctionTransformer* o transformadores personalizados), junto con el preprocesamiento por tipo de dato (*imputación y escalado* para numéricos; *imputación y One-Hot* para categóricas via *ColumnTransformer*), y el estimador final **RandomForestClassifier**. Evalúe con *train/test split* (estratificado si es binario) y reporte métricas (accuracy, precisión, recall, F1, matriz de confusión). Guarde el pipeline final con **joblib** y úselo para inferir sobre el archivo **test_inferencia.csv**.

Parte 3: Red Neuronal MLP (Arquitectura paso a paso)

Diseñe y entrene una MLP sobre el mismo problema. Siga esta guía:
1) **Preparación**: Use el mismo *train/test split* y el mismo preprocesamiento (imputación, escalado, One-Hot).
2) **Arquitectura mínima**: 1 capa oculta (10 neuronas, ReLU); salida sigmoide (binaria).
3) **Arquitectura sugerida**: [n_features → 32 (ReLU) → 16 (ReLU) → 1 (Sigmoid)].
4) **Hiperparámetros**: Adam, lr=1e-3; 50–100 épocas con early stopping (paciencia=5); batch 32–128.
5) **Pérdida/Métricas**: log-loss (binaria), métricas: accuracy, precision, recall, F1.
6) **Evaluación**: comparar con Random Forest en el mismo conjunto de prueba.

```
# ==== Plantilla de arquitectura MLP (scikit-learn) ====
from sklearn.neural_network import MLPClassifier

mlp = MLPClassifier(
    hidden_layer_sizes=(32, 16),
    activation='relu',
    solver='adam',
    learning_rate_init=1e-3,
    max_iter=200,
    random_state=42,
    early_stopping=True,
    n_iter_no_change=5
)
```

```
# NOTA: Preprocéselos igual que en el pipeline (imputación, escalado, One-Hot) antes de entrenar la MLP
# Entrene con X_train, y_train y evalúe con X_test, y_test.
```

Parte 4: Análisis, Conclusiones y Entregables

1) Compare numéricamente RF vs. MLP (mismas métricas/split). 2) Interprete importancias en RF y discuta posibles explicaciones. 3) Explique el impacto de la limpieza. 4) Recomendación final con justificación.
Entregables: (a) notebook/código, (b) pipeline/joblib entrenado, (c) reporte de 1–2 páginas, (d) script de inferencia usando **test_inferencia.csv**.