EDA e Implementación de Modelos de ML

Pregrado en Ciencia de Datos

Asignatura: Machine Learning
Formato único de entrega: Jupyter Book completo, estructurado y autocontenido

Objetivo General

Desarrollar una solución integral y profesional para el problema de clasificación o regresión seleccionado para el proyecto final, aplicando análisis exploratorio, modelado predictivo, técnicas de optimización computacional, evaluación crítica y buenas prácticas de validación e interpretabilidad.

Entregables (todo dentro del Jupyter Book)

1. Análisis Exploratorio de Datos (EDA) Mejorado

Objetivo: profundizar en la comprensión del dataset para fundamentar las decisiones posteriores.

Debe incluir:

- Correcciones y recomendaciones del EDA anterior.
- Visualización e interpretación de:
  - Distribuciones de variables.
  - Outliers.
  - Valores faltantes.
  - Correlaciones y colinealidad.
  - Balance de clases (si aplica).
  - Reducción de dimensionalidad exploratoria (PCA o t-SNE, si corresponde).

2. Modelado Benchmark

Objetivo: implementar y comparar distintos modelos supervisados clásicos, estudiados en el curso, evaluando su desempeño.

Modelos requeridos:

- K-Nearest Neighbors (KNN)
- Naive Bayes

- Regresión Logística (con regularización L1 y L2)
- Ridge
- Lasso
- Árbol de Decisión
- Random Forest
- XGBoost (con interpretabilidad usando LIME)
- Support Vector Machines (SVM)

Cada modelo debe incluir descripción, implementación con código comentado, métricas obtenidas y un análisis interpretativo. Se espera una tabla comparativa al final de esta sección.

3. Técnicas de Balanceo de Clases (para problemas de clasificación)

Técnicas a implementar:

- SMOTE
- ADASYN
- `class_weight='balanced'` (en modelos compatibles)

Se debe analizar el desempeño de los modelos antes y después del balanceo, con especial atención a la métrica de validación seleccionada, ej. Recall, F1 y AUC.

4. Optimización Computacional

Objetivo: acelerar el entrenamiento o predicción sin perder capacidad predictiva, utilizando técnicas especializadas por modelo.

Modelo	**Técnica de Optimización**
KNN	KD-Trees, Ball Trees, o FAISS
Ridge/Lasso	Solver optimizado: saga
Naive Bayes	`partial_fit()` con entrenamiento por lotes
XGBoost	`tree_method='hist'`, `early_stopping_rounds`
SVM	SGDClassifier, LinearSVC o RBF con RFF

Se espera tabla resumen con tiempos, configuraciones y métricas obtenidas.

5. Evaluación de Modelos

Para clasificación:

- Matriz de confusión.
- Curva ROC y valor AUC.
- Tabla de métricas: Accuracy, Precision, Recall, F1, AUC.
- Justificación de la métrica principal elegida.

Para regresión:

- Curva de valores reales vs predichos.
- Tabla de métricas: MAE, MSE, RMSE, R^2.
- Análisis de residuos: ACF, histograma y test de Ljung-Box.

6. Validación Avanzada, Tuning e Interpretabilidad

- **Tuning de hiperparámetros:** uso obligatorio de `sklearn.Pipeline` con GridSearchCV, BayesSearchCV u Optuna.
- **Validación cruzada:** StratifiedKFold (clasificación) o KFold (regresión), mínimo 5 folds.
- **Interpretabilidad:** aplicar en los tres mejores modelos usando `feature_importances_`, coeficientes, SHAP o Permutation Importance.

7. Conclusión Comparativa y Selección del Mejor Modelo

Debe incluir:

- Tabla resumen final con columnas: *Modelo, Métrica Principal, Tiempo, Balanceo, Optimización, Tuning, Interpretabilidad.*
- Justificación crítica del modelo seleccionado como óptimo.
- Discusión sobre ventajas y limitaciones.

Estructura esperada del Jupyter Book

1. Introducción
2. Análisis Exploratorio
3. Modelado Benchmark
4. Técnicas de Balanceo
5. Optimización Computacional

6. Tuning, Validación e Interpretabilidad
7. Evaluación de Modelos
8. Conclusión y Recomendación Final

Rúbrica de Evaluación

Criterio	**Peso (%)**
Análisis Exploratorio	15
Modelado Benchmark	15
Técnicas de Balanceo	10
Optimización Computacional	10
Tuning y Validación Avanzada	20
Interpretabilidad	10
Evaluación crítica y Conclusión Final	10
Claridad, orden y documentación del Jupyter Book	10

Formato de entrega

Se entregará únicamente un Jupyter Book completo, autocontenible, con:

- Notebooks organizados por secciones.
- Código ejecutable y limpio.
- Comentarios y explicaciones claras.
- Visualizaciones integradas.