Evaluación Parcial 3 - Machine Learning

Proyecto Final: Aprendizaje No Supervisado + Integración Completa

---

📋 Información General

Sigla **MLY0100**	**Nombre Asignatura** Machine Learning	**Tiempo Asignado** 4 semanas
✅ ✅ **Proyecto Final + Defensa**	Parejas	Laboratorio / Remoto

% Ponderación: 30% de la asignatura (Evaluación final integradora)

Prerequisitos: Evaluación Parcial 1 (Kedro + CRISP-DM) y Evaluación Parcial 2 (Supervisado + Airflow + DVC + Docker)

tareas en verde: ok

tareas en naranjo: por hacer y dificultad media alta

tareas en rojo: por hacer y dificultad alta

❓ 📝 Descripción de la Evaluación

Esta evaluación parcial representa la culminación del proyecto de Machine Learning iniciado en las evaluaciones anteriores. Los estudiantes deberán integrar y expandir su proyecto existente incorporando técnicas de aprendizaje no supervisado, completando así un pipeline end-to-end de ciencia de datos.

Componentes:

• **Proyecto Práctico Integrado (80%)**

- Implementación de ≥3 técnicas de aprendizaje no supervisado
- Integración con pipelines existentes
- Análisis comparativo y visualización

• Documentación completa

• Defensa Técnica Oral (20%)

• 15-20 minutos de presentación + 10 min preguntas
• Demostración en vivo
• Explicación de decisiones técnicas

Objetivos de Aprendizaje

Al completar esta evaluación final, los estudiantes serán capaces de:

1. 1. Implementar y comparar múltiples algoritmos de aprendizaje no supervisado
2. 2. Integrar técnicas supervisadas y no supervisadas en un proyecto cohesivo 3. 3. Aplicar reducción de dimensionalidad y visualización avanzada 4. 4. Detectar patrones, anomalías y estructuras ocultas en datos
5. 5. Orquestar pipelines complejos de ML con múltiples paradigmas
6. 6. Documentar y comunicar efectivamente resultados técnicos
7. 7. Desplegar y mantener sistemas de ML end-to-end
8. 8. Defender técnicamente decisiones de diseño

Técnicas de Aprendizaje No Supervisado

Los estudiantes deben implementar AL MENOS 3 categorías de técnicas:

1. Clustering (OBLIGATORIO)

Implementar y comparar al menos 3 algoritmos:

• K-Means: Clustering particional basado en centroides
• DBSCAN: Clustering basado en densidad
• Hierarchical Clustering: Clustering jerárquico
• Gaussian Mixture Models: Clustering probabilístico
• OPTICS: Alternativa a DBSCAN para densidad variable

Métricas requeridas: Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz Index, Elbow Method, Dendrogramas

2. Reducción de Dimensionalidad (OBLIGATORIO)

Implementar al menos 2 técnicas:

• PCA: Análisis de componentes principales (varianza explicada, loadings, biplot) •
t-SNE: Visualización 2D/3D de alta dimensión
• UMAP: Alternativa moderna a t-SNE
• SVD/Truncated SVD: Para datos sparse

3. Detección de Anomalías (OPCIONAL)

• Isolation Forest
• Local Outlier Factor (LOF)
• One-Class SVM
• Autoencoders

4. Reglas de Asociación (OPCIONAL)

• Apriori Algorithm
• FP-Growth
• Métricas: support, confidence, lift

⚙ Requisitos Técnicos de Integración

Arquitectura del Proyecto Final

Framework Kedro:

• Pipeline unsupervised_learning/ integrado
• Catálogo actualizado
• Parámetros configurables

Orquestación Airflow:

• DAG: data_engineering → supervised → unsupervised
• Tasks independientes por algoritmo
• Manejo de dependencias

Versionado DVC:

• Features de clustering versionadas
• Modelos de reducción dimensional
• Métricas de experimentos

(

Docker:

- Dockerfile actualizado
- docker-compose.yml completo (crear un nuevo compose multicontenedor)
- Volúmenes configurados (?)

Librerías Adicionales

- scikit-learn>=1.3.0
- plotly>=5.0.0, seaborn>=0.12.0, matplotlib>=3.7.0
- umap-learn>=0.5.0
- pyod>=1.1.0 (anomalías)
- mlxtend>=0.22.0 (reglas asociación)
- hdbscan>=0.8.0
- shap>=0.42.0 (interpretabilidad)

Estructura del Proyecto Actualizada

```
proyecto-ml-final/
├── conf/
│   ├── base/
│   │   ├── catalog.yml          # ← ACTUALIZADO
│   │   └── parameters.yml       # ← ACTUALIZADO
│   └── local/
├── data/
│   ├── 01_raw/ ... 05_model_input/
│   ├── 06_models/               # ← NUEVO
│   ├── 07_model_output/         # ← NUEVO
│   └── 08_reporting/            # ← NUEVO
├── dags/
│   └── ml_pipeline_master.py    # ← ACTUALIZADO
├── src/proyecto_ml/pipelines/
│   ├── data_engineering/
│   ├── supervised_learning/
│   │   ├── classification/
│   │   └── regression/
│   ├── unsupervised_learning/   # ← NUEVO
│   │   ├── clustering/
│   │   ├── dimensionality_reduction/
│   │   ├── anomaly_detection/
│   │   └── association_rules/
│   └── reporting/
├── notebooks/
```

```
│   ├── 01-04: De EP1 y EP2
│   ├── 05_unsupervised_learning.ipynb  # ← NUEVO
│   └── 06_final_analysis.ipynb         # ← NUEVO
├── docs/
│   ├── architecture.md         # ← NUEVO
│   └── unsupervised_analysis.md # ← NUEVO
├── docker/
│   ├── Dockerfile              # ← ACTUALIZADO
│   └── docker-compose.yml      # ← ACTUALIZADO
└── README.md                   # ← ACTUALIZADO
```

_____z_____

◆◆ Rúbrica de Evaluación (80% Práctica)

Escala de Evaluación

Nivel	% Logro	Nota	Descripción
Muy buen desempeño	100%	7.0	Excelencia completa
Buen desempeño	80%	5.6	Alto desempeño
Desempeño aceptable	60%	4.2	Elementos básicos
Desempeño incipiente	40%	2.8	Omisiones importantes
Desempeño insuficiente	20%	1.4	Incorrecto
No logrado	0%	1.0	No cumple mínimos

Indicadores de Evaluación (10 × 8% = 80%)

1. Clustering (8%)

100%: ≥3 algoritmos, métricas completas, análisis óptimo de K, visualizaciones profesionales

80%: 3 algoritmos correctos, buenas métricas y visualizaciones

60%: 2-3 algoritmos funcionales, métricas básicas

40%: 1-2 algoritmos con errores

20%: Implementación deficiente

2. Reducción Dimensional (8%)

100%: PCA completo (varianza, loadings, biplot) + t-SNE/UMAP con múltiples parámetros, visualizaciones interactivas

80%: PCA y t-SNE/UMAP implementados, buenas visualizaciones

60%: PCA funcional + una técnica adicional

40%: Solo PCA básico o con errores

20%: No implementa correctamente

3. Integración con Supervisados (8%)

100%: Clustering como feature engineering para supervisados, análisis de mejora, pipeline unificado?

60%: Integración básica funcional
20%: No hay integración efectiva

4. Análisis de Patrones (8%)

100%: Análisis profundo por cluster: estadísticas, perfiles, características, interpretación de negocio, etiquetado semántico

60%: Análisis básico de estadísticas por cluster

5. Orquestación Airflow (8%)

100%: DAG maestro complejo, dependencias correctas, parametrizable, manejo de errores, logs, XComs

60%: DAG básico funcional

6. Versionado DVC (8%)

100%: DVC versiona todos los artefactos, métricas trackeadas, .dvc files correctos, dvc.yaml con etapas

60%: DVC funcional versionando elementos básicos

7. Dockerización (8%)

100%: Dockerfile multi-stage optimizado, docker-compose con servicios completos, volúmenes, documentación

60%: Dockerfile básico funcional

8. Técnicas Adicionales (8%)

100%: Detección de anomalías con ≥2 algoritmos, análisis de outliers O reglas de asociación completas

NOTA: Si no implementan, este 8% se redistribuye

9. Documentación (8%)

100%: README excepcional, notebooks con narrativa profesional, visualizaciones interactivas, docstrings completos

60%: Documentación básica funcional

10. Innovación (8%)

100%: AutoML, ensemble avanzado, APIs, monitoring, A/B testing, SHAP

avanzado **60%:** 1 elemento adicional de valor

Defensa Técnica Oral (20%)

Formato

Duración **Modalidad**	**15-20 min presentación + 10 min preguntas** Presencial o remota
Materiales	Presentación + demo en vivo
Evaluación	Individual (ambos deben demostrar conocimiento)

Estructura de Presentación

• 1-2 min: Contexto y objetivos
• 2-3 min: Arquitectura y decisiones de diseño
• 3-4 min: Pipeline de datos y feature engineering
• 4-5 min: Modelos supervisados (resultados EP2)
• 5-7 min: Análisis no supervisado (clustering, dimensionalidad, insights) •
2-3 min: Integración, orquestación y despliegue
• 1-2 min: Desafíos y soluciones
• 1-2 min: Conclusiones y trabajo futuro

Criterios Defensa (20%)

Dominio técnico (40%): Comprensión algoritmos, explicación decisiones, métricas, respuesta preguntas

Comunicación (30%): Claridad, terminología, visualizaciones, gestión

tiempo **Demo práctica (20%):** Ejecución en vivo, navegación código, outputs

Trabajo en equipo (10%): Ambos demuestran conocimiento, coordinación

Preguntas Tipo

• ¿Por qué K-Means sobre DBSCAN?
• ¿Cómo determinaron K óptimo?
• ¿Impacto de reducción dimensional en supervisados?
• ¿Interpretación del PC1 en PCA?
• ¿Cómo gestionan drift de datos?
• ¿Cómo escalaría a 100x más datos?
• Explique integración Airflow + Kedro
• ¿Estrategia de versionado de experimentos?

Entregables

Obligatorios

✅ ☑ Repositorio Git con proyecto Kedro completo

✅ ☑ ≥3 algoritmos clustering + ≥2 reducción dimensional

✅ ☑ DAGs Airflow operativos

✅ ☑ DVC versionando artefactos

✅ ☑ Docker funcional

✅ ☐ 6+ notebooks de análisis

✅ ☐ README completo

✅ ☐ Documentación técnica

✅ ☐ Reporte comparativo

✅ ☐ Presentación defensa

Opcionales (+0.5 puntos)

⭐ ☐ Tests unitarios (pytest)

⭐ ☐ CI/CD (GitHub Actions)

⭐ ☐ API REST (FastAPI)

⭐ ☐ Dashboard (Streamlit/Dash)

⭐ ☐ MLflow tracking

⭐ ☐ SHAP explainability

⭐ ☐ Deployment cloud

☐ Cronograma Sugerido

Semana	Actividades	Entregables
Semana 1	Clustering + métricas	Pipeline clustering + notebook
Semana 2	Reducción dimensional + viz	Pipeline dim_reduction + visualizaciones
Semana 3	Integración + Airflow + Docker	Proyecto integrado completo
Semana 4	Documentación + presentación + defensa	Proyecto final + defensa

---

✅ ☐ Checklist de Entrega

Verificar antes de entregar:

Código:

- [ ] kedro run sin errores
- [ ] ≥3 clustering + ≥2 dim reduction

- [ ] Integración con supervisados funciona
- [ ] Docstrings y comentarios
- [ ] Respeta PEP8

Orquestación:

- [ ] Airflow DAG funciona completo
- [ ] DVC versiona todo
- [ ] Docker build correcto
- [ ] docker-compose up levanta servicios
- [ ] Reproducible en otro equipo

Documentación:

- [ok] 6+ notebooks documentados
- [ok] README completo
- [ok] Docs técnicos
- [ok] Reporte comparativo
- [ ] Visualizaciones profesionales

Calidad:

- [ ] requirements.txt actualizado
- [ok] .gitignore correcto
- [ok] Sin datos sensibles
- [ok] Commits descriptivos
- [ok] Sin archivos innecesarios

Defensa:

- [ ] Presentación lista (15-20 slides)
- [ ] Demo testeada
- [ok] Ambos preparados
- [ok] Respuestas preparadas

?? Recursos de Apoyo

Documentación Oficial

• Scikit-learn Clustering: https://scikit-learn.org/stable/modules/clustering.html • Scikit-learn Decomposition: https://scikit
learn.org/stable/modules/decomposition.html

• UMAP: https://umap-learn.readthedocs.io/
• Kedro: https://kedro.readthedocs.io/
• Airflow: https://airflow.apache.org/docs/
• DVC: https://dvc.org/doc

⚠ Consideraciones Importantes

Integridad Académica: Ambos deben contribuir equitativamente. Uso de IA (ChatGPT) permitido pero deben entender todo el código.

Gestión del Tiempo: 4 semanas es ajustado. Distribuir trabajo y comenzar temprano.

Comunicación: Avances semanales esperados. Usar laboratorios para dudas.

Realismo: Proyecto realista. No comprometer calidad por cantidad.

_

❓ Criterios de Éxito

9. 1. Ejecutarse sin errores en Docker
10. 2. Comprensión de técnicas supervisadas y no supervisadas
11. 3. Integración coherente de todos los componentes
12. 4. Insights accionables del análisis
13. 5. Profesionalmente documentado
14. 6. Defender técnicamente decisiones
15. 7. Proyecto de portfolio profesional

¡Éxito en su proyecto final! ❓

Este es el momento de demostrar todo lo aprendido. Un proyecto bien ejecutado será una pieza clave de su portfolio profesional. Disfruten el proceso de descubrir patrones ocultos y construir sistemas ML end-to-end.

Recuerden: La excelencia está en los detalles. Código limpio y bien documentado vale tanto como los resultados técnicos