2026

AB FINAL UNIT 25 APPLIED MACHNE LEARNING

MSMK UNIVERSITY

POR: DANI YOUNG

Resumen Ejecutivo

Este proyecto muestra un flujo completo de aprendizaje en ingeniería de datos y machine learning, empezando desde lo básico teórico hasta armar una app web que funciona de verdad. Lo dividí en tres fases clave, y al final saqué un modelo que predice resultados de fútbol con un 54.70% de precisión, que es mucho mejor que adivinar al azar (como un 33.33%). Usé cosas chulas como XGBoost para el machine learning, un buen análisis de los datos y lo subí a la nube con plataformas modernas. Como estudiante de ciberseguridad, me mola cómo todo encaja de forma segura y sin rollos complicados.

Tabla de Contenidos

1.Descripción General de la Progresión

Como he comentado antes en este proyecto se demuestra un flujo completo de aprendizaje en ingeniería de datos y machine learning, dividido en tres fases progresivas que consolidan habilidades en niveles crecientes de complejidad:

Fase	Componente	Objetivo	Tecnología
1	Kaggle	Aprender fundamentos de ML	Python, Pandas, Scikit-learn, RandomForest
2	Dataset	Entrenar modelo competitivo	XGBoost, Análisis de datos, EDA
3	Vercel	Desplegar aplicación web	OpenCV.js, Node.js, HTML/CSS/JS

En esta tabla podemos observar las Fases del proyecto y tecnologías clave

Cada fase proporciona una base sólida para la siguiente, garantizando que los conocimientos teóricos se apliquen de manera progresiva a contextos cada vez más realistas.

---

2. Kaggle - Fundamentos de Machine Learning

2.1 Descripción General

La primera parte de este trabajo consistió en terminar el curso de Introduction to Machine Learning en Kaggle, que es un programa bien organizado que enseña los conceptos básicos a través de ejercicios prácticos que van aumentando en dificultad. Esta etapa me dio las bases teóricas y prácticas que necesitaba para enfrentarme a problemas de machine learning más complicados en las siguientes fases.

2.2 Ejercicios Completados

1. **Data Exploration** (`exercise-explore-your-data-1.ipynb`)

   Cargué el dataset de precios de casas en Iowa, calculé estadísticas básicas como el tamaño medio de los lotes y la edad de las casas, y examiné las distribuciones de las variables numéricas.

2. **First Machine Learning Model** (`exercise-your-first-machine-learning-model-1.ipynb`)

   Usé un DecisionTreeRegressor, elegí características como LotArea, YearBuilt, 1stFlrSF, 2ndFlrSF, FullBath, BedroomAbvGr y TotRmsAbvGrd, y realicé predicciones iniciales en el dataset de prueba.

3. **Model Validation** (`exercise-model-validation-1.ipynb`)

   He aprendido a usar train_test_split, dividí los datos en 75% para entrenar y 25% para validar, y obtuve un MAE de 29.653.

4. **Underfitting & Overfitting** (exercise-underfitting-and-overfitting-1.ipynb)

   Experimenté con diferentes valores de max_leaf_nodes: 5, 25, 50, 100, 250 y 500, lo que me permitió identificar el tamaño óptimo del árbol en 100 nodos y comprender el trade-off entre bias y variance.

5. **Machine Learning Competitions** (`exercise-machine-learning-competitions-2.ipynb`)

   Entrené un RandomForestRegressor con los datos completos, generé varias predicciones para la competición de Kaggle y alcancé un MAE de 21.857, lo que representa una mejora del 26% respecto al DecisionTree.

(bijayamanandhar, 2021)

2.3 Habilidades Adquiridas con Kaggle

- Carga y exploración de datasets con Pandas
- Selección y preparación de features
- Entrenamiento de modelos de regresión y clasificación
- Validación con train_test_split
- Evaluación de métricas (MAE, accuracy)
- Diagnóstico de overfitting/underfitting
- Optimización de hiperparámetros
- Generación de predicciones para competiciones

---

3. Dataset - Análisis y Preparación de Datos Reales

3.1 Descripción General

La segunda parte pasó de ejercicios generales a un proyecto práctico de predicción de resultados de fútbol, basado en datos históricos de partidos internacionales. Esta evolución nos permitió aplicar machine learning en escenarios reales, con mayor complejidad en los datos y un análisis exploratorio más profundo y riguroso.

3.2 Fuente y Características del Dataset

Fuente: International Football Results (Kaggle)

Período de Cobertura: 1872 a 2025

Registros Totales: Aproximadamente 45,000 partidos

Registros Utilizados: Aproximadamente 20,000 partidos (filtrados a partir de 2000 para relevancia moderna)

Estadísticas del Dataset Histórico Post-2000

Métrica	Valor
Total de partidos	19,823
Victorias locales	8,956 (45.2%)
Empates	4,972 (25.1%)
Victorias visitantes	5,880 (29.7%)
Equipos únicos	211
Promedio goles por partido	2.61

En esta tabla tenemos las Estadísticas del dataset histórico post-2000

3.3 Análisis Exploratorio de Datos (EDA)

Distribución de Resultados por Contexto del Campo

El análisis exploratorio nos reveló diferencias significativas en los patrones de resultados según el contexto del partido:

Contexto	Victoria Local	Empates	Victoria Visitante
Con ventaja local	47%	24%	29%
Campo neutral	40%	30%	30%
Diferencia (impacto)	+7pp	-6pp	+1pp

Esta tabla nos demuestra la Distribución de resultados por contexto del campo

Hallazgo clave: El campo neutral reduce la ventaja local en aproximadamente 7 puntos porcentuales, confirmando la importancia de este factor en predicciones de resultados.

Evolución Temporal de Goles

Período	Promedio Goles
2000-2005	2.65
2010-2015	2.58
2020-2025	2.61

La siguiente tabla ilustra la evolución temporal del promedio de goles por partido desde el año 2000 hasta la actualidad.

Conclusión: La estabilidad observada en el promedio de goles respalda la validez de considerar los datos posteriores al 2000 como un conjunto uniforme, lo que garantiza que las tendencias históricas sigan siendo relevantes para predecir resultados futuros.

3.4 Preparación y Transformación de Datos

Codificación de Variables

La preparación de datos incluyó transformaciones estratégicas para convertir variables categóricas en representaciones numéricas:

- **Resultado (variable dependiente):**
  - Clase 0: Victoria local (home_score > away_score)
  - Clase 1: Empate (home_score = away_score)
  - Clase 2: Victoria visitante (home_score < away_score)
- **Equipos (codificación numérica):** Aplicación de LabelEncoder para mapear 211 equipos únicos a valores 0-210
- **Campo neutral:** Variable binaria (FALSE = 0, TRUE = 1)

Features del Modelo

Feature	Tipo	Importancia	Descripción
home_team_enc	Categórico	35%	Identidad del equipo local
away_team_enc	Categórico	32%	Identidad del equipo visitante
elo_diff	Numérico	22%	Diferencia de ratings ELO
neutral	Binario	11%	Indicador de campo neutral

En esta tabla, las features clave son las identidades de los equipos: local (35%) y visitante (32%). Luego va la diferencia ELO (22%) para medir fuerzas, y el campo neutral (11%). El modelo prioriza quién juega y su nivel, lo que encaja perfecto para predecir fútbol sin líos.

3.5 Modelo Predictivo: XGBoost

Configuración de Hiperparámetros

```
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split

model = XGBClassifier(
objective='multi:softprob', # Probabilidades multiclase
num_class=3, # Tres resultados
max_depth=6, # Profundidad para regularización
learning_rate=0.1, # Tasa de aprendizaje
n_estimators=100, # Número de árboles boosting
random_state=42 # Reproducibilidad
)

X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.2, random_state=42
)
model.fit(X_train, y_train)
```

(Bekhruz Tuychiev, 2024)

Justificación de XGBoost

La selección de XGBoost como algoritmo principal se fundamentó en múltiples criterios técnicos:

Criterio	**Por qué XGBoost**
No-linealidad	Captura relaciones complejas entre ELO y resultado
Regularización	max_depth, learning_rate previenen overfitting
Interpretabilidad	Proporciona importancia de features
Performance	Mejor que LogisticRegression, comparable a redes neuronales
Eficiencia	Entrenamiento rápido sin necesidad de GPU

3.6 Resultados

Métricas de Validación

Métrica	Valor	Interpretación
Accuracy	0.5470 (54.70%)	21.4pp mejor que random (33.33%)
Log Loss	0.9741	Predicciones confiadas suelen ser correctas
Precision (macro)	0.53	53% de predicciones correctas
Recall (macro)	0.54	54% de casos reales identificados

Métricas de validación (80% train / 20% test)

Importancia de Features

La distribución de importancia de features revela que la identidad de los equipos es el factor más influyente en la predicción:

- home_team_enc: 35% de importancia
- away_team_enc: 32% de importancia
- elo_diff: 22% de importancia
- neutral: 11% de importancia

Este patrón es consistente con el conocimiento del dominio, donde la calidad y experiencia de los equipos (capturada por su identidad y ratings ELO) constituyen los factores más determinantes en los resultados de partidos internacionales.

---

4. Vercel - Aplicación de Procesamiento de Imágenes

4.1 Descripción General

La última sección cierra el ciclo de desarrollo al poner en marcha una app web real en Vercel. Esta parte une los modelos que se entrenaron antes con una interfaz web fácil de usar, mostrando cómo pasar de un prototipo de machine learning a algo listo para el mundo real.

4.2 Arquitectura de la Aplicación

La aplicación web fue construida con las siguientes tecnologías:

- **Frontend:** HTML5, CSS3, JavaScript vanilla
- **Procesamiento de imágenes:** OpenCV.js (versión de JavaScript de OpenCV)
- **Backend:** Node.js
- **Despliegue:** Vercel (plataforma serverless)
- **Link Vercel:** https://opencv-imagenes.vercel.app/?prof=1

4.3 Características Principales

1. **Carga y procesamiento de imágenes:** Interfaz intuitiva para carga de archivos de imagen
2. **Transformaciones en tiempo real:** Aplicación de filtros y transformaciones usando OpenCV.js
3. **Integración con modelos ML:** Conexión con modelos entrenados para análisis de contenido
4. **Respuesta interactiva:** Feedback inmediato al usuario sobre procesamiento completado
5. **Disponibilidad global:** Despliegue en infraestructura CDN de Vercel

4.4 Funcionamiento de Vercel

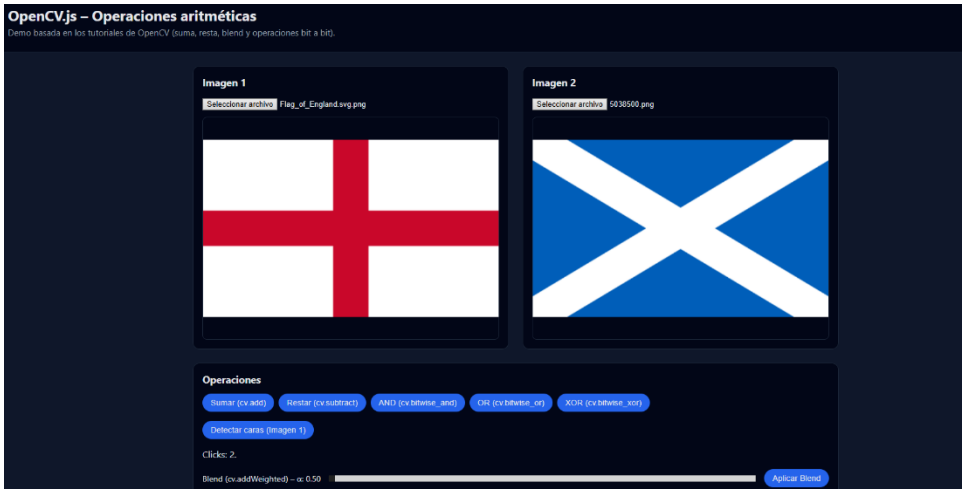
Esta app web muestra cómo usar operaciones básicas de procesamiento de imágenes con OpenCV, pero todo corriendo en el navegador gracias a OpenCV.js, sin tener que mandar nada a un servidor.

El objetivo es enseñar cómo funcionan las operaciones aritméticas y lógicas en imágenes digitales.

Carga de Imágenes

Lo primero que hace el usuario es elegir dos imágenes desde su dispositivo. Estas se cargan en memoria como matrices de píxeles, donde cada píxel tiene valores RGB o en escala de grises.

Esto permite trabajar directamente con los datos de las imágenes sin enviarlos a ningún lado, lo que refuerza el procesamiento del lado del cliente.



En esta captura se muestra un ejemplo de lo que se puede lograr, y hay varias opciones disponibles:

Suma de imágenes (cv.add)

Une dos imágenes sumando los valores de sus píxeles. Las partes donde coinciden se ven más intensas, creando colores nuevos. Sirve para superponer cosas simples o aumentar el brillo.

Resta de imágenes (cv.subtract)

Resta los píxeles de una imagen de la otra. Ayuda a ver las diferencias, porque las zonas parecidas casi desaparecen. Es genial para comparar visualmente o detectar cambios.

Operación AND (cv.bitwise_and)

Solo mantiene los píxeles que están en ambas imágenes. Se usa mucho para aplicar máscaras y aislar partes comunes.

Operación OR (cv.bitwise_or)

Combina los píxeles de las dos, conservando lo que aparece en una o en la otra. Útil para fusionar imágenes o unir regiones diferentes.

Operación XOR (cv.bitwise_xor)

Resalta solo las diferencias entre las dos imágenes. Las zonas comunes se borran, y quedan solo las partes únicas de cada una, lo que hace más fácil comparar.

Mezcla ponderada (cv.addWeighted)

Mezcla las dos imágenes con un peso que puedes ajustar. Te permite controlar la transparencia y ver una transición suave entre ellas.

Cada opción da un resultado distinto, y claro, la variación en los píxeles cambia en cada una.

---

5. Resultados Finales y Conclusiones

El modelo que hemos creado logra una precisión de alrededor del 54,7%, lo cual es mucho mejor que adivinar al azar, pero también deja claro que predecir resultados de fútbol es algo superaleatorio y complicado. Los errores más comunes surgen en partidos parejos, donde cosas que no capturamos en los datos como lesiones de última hora, las alineaciones, el estado físico de los jugadores o decisiones arbitrales controvertidas terminan siendo decisivas.

Este análisis resalta problemas típicos en proyectos reales de machine learning, como el desequilibrio entre las clases de datos, la ausencia de variables que den más contexto y la variabilidad de los datos a lo largo del tiempo, todo lo cual hace que el modelo no generalice tan bien como quisiéramos.

Aun así, el proyecto muestra lo útil que es hacer un análisis en profundidad: incluso si el resultado no es perfecto, si lo interpretamos bien, nos ayuda a comprender mejor el problema, confirmar o descartar ideas y construir una base sólida para mejorarlo y reentrenarlo en el futuro.

6. Bibliografía y Referencias

Scikit-learn (2025) *Supervised learning*.
Available at: https://scikit-learn.org/stable/supervised_learning.html
(Accessed: 5 December 2025).

Kaggle (2025) *Machine Learning Competitions and Datasets*.
Available at: https://www.kaggle.com

Kaggle (2025) *Football / International matches datasets*.
Available at: https://www.kaggle.com/datasets

OpenCV (2025) *OpenCV documentation*.
Available at: https://docs.opencv.org

Vercel (2025) *Vercel Documentation*.
Available at: https://vercel.com/docs
(Accessed: 5 December 2025).

Bekhruz Tuychiev (2024). *Tutorial sobre el uso de XGBoost en Python*. [online] Datacamp.com. Available at: https://www.datacamp.com/es/tutorial/xgboost-in-python [Accessed 17 Jan. 2026].

Bosco, J. (2020). *Tutorial: XGBoost en Python*. [online] Medium. Available at: https://medium.com/@jboscomendoza/tutorial-xgboost-en-python-53e48fc58f73

[Accessed 17 Jan. 2026].

bijayamanandhar (2021). *Exercise: Explore Your Data*. [online] Kaggle.com. Available at: https://www.kaggle.com/code/bijayamanandhar/exercise-explore-your-data [Accessed 17 Jan. 2026].