FACULTAD DE CIENCIAS MATEMÁTICAS

Y FÍSICAS

Carrera: Ciencia de Datos e Inteligencia

Artificial

PROYECTO FINAL

ALMACENES DE DATOS Y MINERIA DE DATOS

Curso:

CDDEIA-ELNO-5-2

Estudiante:

CARLOS BURGOS JEAN CARLOS

Maestro:

ING, LEON GRANIZO OSCAR DARIO

REPOSITORIO GITHUB

https://github.com/naejcbk/Predicci-n-de-Deserci-n-Estudiantil---UG-Student-Dropout-Prediction-System---University-of-Guayaquil

Fase 1: Comprensión del Negocio.

El Problema: La deserción estudiantil afecta la estabilidad de la institución y el futuro de los alumnos. Necesitamos pasar de un modelo reactivo (ver quién se fue) a uno proactivo (predecir quién se irá).

Objetivo del Proyecto:

Desarrollar un modelo de clasificación binaria (Riesgo vs. No Riesgo) basado en el historial académico.

Identificar al menos al 75% de los desertores reales (Recall) para permitir una intervención administrativa oportuna.

Fase 2: Entendimiento de los Datos

Volumen de datos: 4,448 registros.

Población total: 488 estudiantes.

Variables Identificadas.

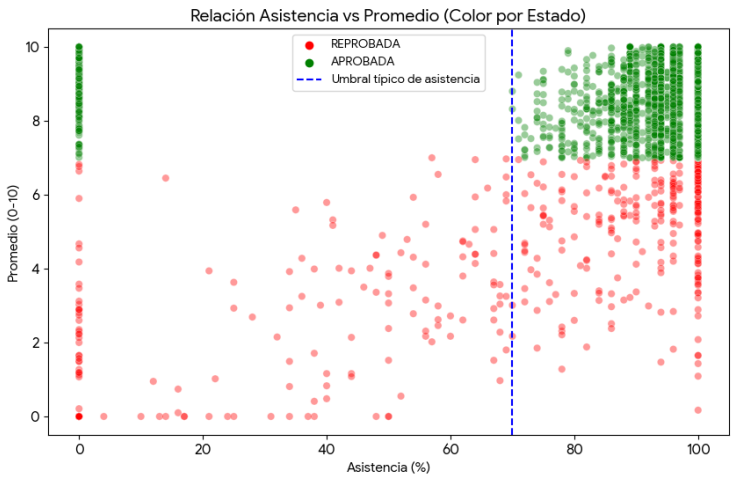
PROMEDIO: Nota numérica de la materia. (Sobre 10)

ASISTENCIA: Porcentaje de asistencia a clase.

NIVEL: Semestre en el que se encuentra la materia que cursa el estudiante (1 al 4).

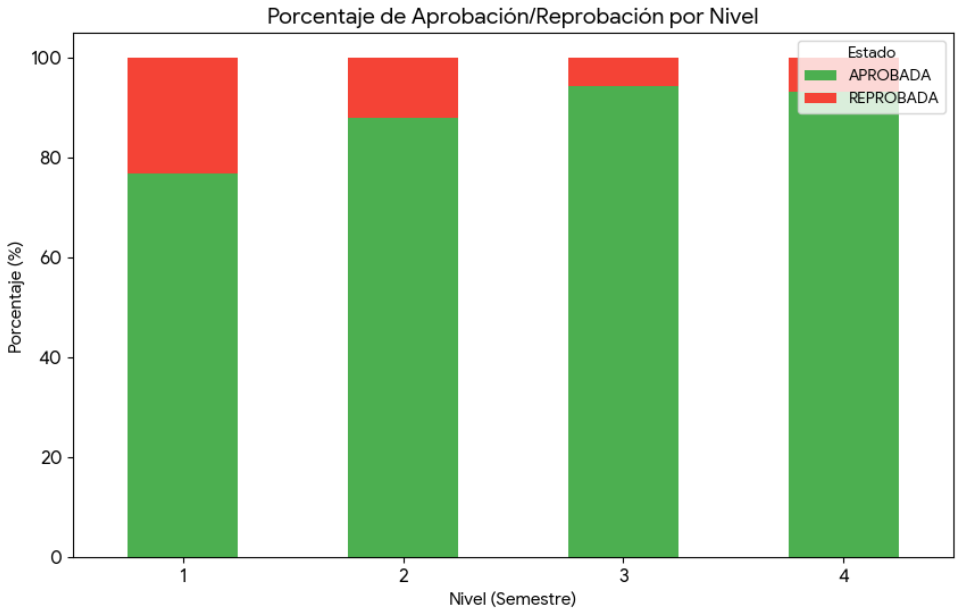
ANÁLISIS DE VARIABLES

ASISTENCIA VS PROMEDIO



- Para aprobar una materia es necesario el 70% de asistencia. Los puntos rojos (reprobados) aumentan masivamente a la izquierda de esa línea.
- Muchos estudiantes tienen 0% de asistencia y 0 de promedio. Estos son los casos de "deserción silenciosa" que el modelo debe detectar antes de que ocurran.
- La mayor cantidad de puntos se concentran por debajo de la nota 7.0, pero hay un grupo enorme en la nota 0.0. Un promedio de 0.0 no suele ser falta de capacidad, sino abandono. Es la señal más clara de deserción temprana.

NIVEL (SEMESTRE)



El gráfico de barras por nivel muestra que el Nivel 1 tiene el mayor porcentaje de materias reprobadas (la barra roja es más alta proporcionalmente). Podemos inferir que la deserción es un problema de los primeros semestres. Si un estudiante supera el Nivel 2, su probabilidad de quedarse aumenta.

DEFINICION DE VARIABLE OBJETIVO

Definición de "Deserción": Para el desarrollo del modelo predictivo, se definió la variable objetivo bajo un criterio de **ausencia en el periodo vigente**. Dado que los datos abarcan hasta el ciclo 2025-2026, se etiquetó como 'Desertor' a todo estudiante con historial previo que no registró matrícula en el ciclo actual. Este enfoque permite al modelo aprender los patrones de rendimiento y asistencia que preceden al abandono definitivo.

Fase 3: Preparación de los Datos

Esta fase transformó los **4,448 registros transaccionales** en una tabla maestra de **488 estudiantes** optimizada para la IA.

```python
import pandas as pd
import numpy as np

# Configuración del archivo original
archivo_ug = 'REPORTE_RECORD_ESTUDIANTIL_ANONIMIZADO.xlsx'

try:
    # Usamos engine='openpyxl' por seguridad con archivos .xlsx modernos
    df = pd.read_excel(archivo_ug, sheet_name='Sheet1')
    print("Archivo UG cargado correctamente.")
except Exception as e:
    print(f"Error al cargar el Excel: {e}")
    exit()

# FUNCIÓN CORREGIDA: Convierte las comas en puntos
def corregir_notas(v):
    if isinstance(v, str):
        return float(v.replace(',', '.'))
    return float(v)

df['nota_limpia'] = df['PROMEDIO'].apply(corregir_notas)

# Ajuste de asistencia por convalidaciones/movilidad
convalida_mask = df['GRUPO/PARALELO'].str.contains('MOVILIDAD|CONVALIDACION', na=False, case=False)
df['asistencia_final'] = df['ASISTENCIA']
df.loc[convalida_mask & (df['ESTADO'] == 'APROBADA'), 'asistencia_final'] = 100

# Identificar estudiantes activos en el ciclo 2025
activos_2025 = df[df['PERIODO'].str.contains('2025 - 2026', na=False)]['ESTUDIANTE'].unique()

# CREACIÓN DE LA TABLA MAESTRA
# Agrupamos por estudiante con los 3 nombres finales
dataset = df.groupby('ESTUDIANTE').agg(
    promediohistorico=('nota_limpia', 'mean'),
    promedioasistencia=('asistencia_final', 'mean'),
    **{'# de Semestre': ('NIVEL', 'max')}
).reset_index()

# LÓGICA DE VARIABLE OBJETIVO (DESERTOR)
# 1. Por ausencia en 2025
dataset['Desertor'] = dataset['ESTUDIANTE'].apply(lambda x: 0 if x in activos_2025 else 1)

# 2. Por reglamento: 3ra matrícula reprobada
perdio_3ra = df[(df['NO. VEZ'] == 3) & (df['ESTADO'] == 'REPROBADA')]['ESTUDIANTE'].unique()
dataset.loc[dataset['ESTUDIANTE'].isin(perdio_3ra), 'Desertor'] = 1

# 3. Por reglamento: Asistencia insuficiente (< 70)
dataset.loc[dataset['promedioasistencia'] < 70, 'Desertor'] = 1

# Guardar resultado
dataset.to_csv('dataset.csv', index=False)
print("✅ 'dataset.csv' generado con éxito.")
```

- **Saneamiento de Formatos:** Se implementó una limpieza para sustituir comas por puntos en el récord académico, forzando la conversión de cadenas de texto a tipos flotantes (float64).
- **Agregación de Datos:** Se realizó un proceso de groupby por estudiante para consolidar el historial en indicadores únicos: Promedio Histórico, Promedio de Asistencia y Nivel Actual.
- **Normalización y Etiquetado:** Se corrigieron casos de movilidad y se generó la variable objetivo (**Desertor**) identificando a quienes no registraron matrícula en el ciclo vigente o incumplieron el límite legal del 70% de asistencia.
- **División Estratificada:** Se utilizó un esquema de 80% para entrenamiento y 20% para pruebas, aplicando un muestreo estratificado para mantener la proporción de desertores en ambos conjuntos.

Fase 4: Modelado

Se seleccionó el algoritmo **Random Forest Classifier** por su robustez ante datos institucionales.

```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import joblib

# 1. CARGA DE DATOS
# Se utiliza la librería pandas para leer el archivo CSV que contiene el historial académico.
df = pd.read_csv('dataset.csv')

# 2. DEFINICIÓN DE VARIABLES (Features y Target)
# X: Son las características que el modelo usará para aprender (promedio, asistencia y semestre).
X = df[['promediohistorico', 'promedioasistencia', '# de Semestre']]
# y: Es el "objetivo" o la respuesta que queremos predecir (si desertó o no).
y = df['Desertor']

# 3. DIVISIÓN DEL DATASET
# Se separan los datos en dos grupos: uno para que el modelo estudie (train) y otro para examinarlo
(test).
# test_size=0.2: El 20% de los datos se reserva para la evaluación final.
# stratify=y: Asegura que la proporción de desertores y no desertores sea igual en ambos grupos.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

# 4. CONFIGURACIÓN DEL MODELO (Bosque Aleatorio)
# Se crea el clasificador con parámetros específicos para evitar errores de predicción:
modelo = RandomForestClassifier(
    n_estimators=100,          # Crea 100 árboles de decisión para votar por el resultado.
    max_depth=5,               # Limita la profundidad de los árboles para evitar que memoricen los datos
(overfitting).
    min_samples_leaf=10,       # Pide que al menos 10 casos coincidan para crear una "hoja" de decisión.
    random_state=42,           # Asegura que el entrenamiento sea reproducible y no cambie al azar.
    class_weight='balanced'    # Crucial: Da más peso a los casos de deserción para compensar si hay
pocos en el CSV.
)

# 5. ENTRENAMIENTO
# El modelo busca patrones y relaciones entre las notas/asistencia y la probabilidad de abandonar.
modelo.fit(X_train, y_train)

# 6. EXPORTACIÓN DEL MODELO
# Se guarda el modelo entrenado en un archivo físico (.pkl).
# Esto permite que la aplicación de Streamlit pueda usarlo después sin tener que volver a entrenar.
joblib.dump(modelo, 'modelo_desercion.pkl')

print("Modelo de IA entrenado y guardado como 'modelo_desercion.pkl'.")
```

- **Configuración del Modelo:** El bosque se configuró con 100 árboles de decisión, controlando la profundidad (max_depth=5) para evitar el sobreajuste y utilizando pesos balanceados (class_weight='balanced') para dar prioridad a la detección de la clase minoritaria (desertores).
- **Jerarquía de Predictores:** El análisis de importancia determinó que el Promedio Histórico es el factor de mayor peso en la predicción, seguido por la Asistencia y el Semestre.
- **Lógica de Clasificación:** El modelo evalúa simultáneamente las tres dimensiones; por ejemplo, identifica como "Riesgo Alto" a perfiles con promedios bajos en semestres iniciales, incluso si su asistencia es regular.

Fase 5: Evaluación

Los resultados confirman que el modelo supera los objetivos planteados inicialmente.

Métricas Obtenidas:

- Exactitud (Accuracy): 84% de acierto general.
- Sensibilidad (Recall): 86%, superando la meta del 75% establecida en la Fase 1. Esto garantiza que de cada 100 desertores reales, el sistema detecta a 86 proactivamente.
- Análisis de Errores: Se identificó que es preferible asumir el costo de "Falsos Positivos" (tutorías preventivas innecesarias) que el riesgo de "Falsos Negativos" (perder a un estudiante no detectado).
- Hallazgos Clave: La zona de riesgo crítico se sitúa bajo el promedio de 7.0, y el Nivel 1 se confirma como el punto de mayor vulnerabilidad académica.

Fase 6: Despliegue

La solución trasciende el código para convertirse en una estrategia institucional.

Simulador Operativo: Se desarrolló una interfaz en Streamlit para que Bienestar Estudiantil realice diagnósticos individuales con umbrales calibrados:

- **Riesgo Alto (>60%):** Derivación inmediata a intervención presencial.
- **Riesgo Medio (>30%):** Monitoreo preventivo de asistencia.

Plan de Integración: Se propone integrar este motor como un módulo del portal académico de la UG para generar reportes automáticos al cierre de cada parcial.

Sostenibilidad: Se entrega documentación técnica para la recalibración periódica del modelo conforme evolucionen los patrones de conducta de la carrera.