Predicción de asaltos por colonia, mes, día y turno{

Francisco Antonio López Ricardez
Emilio Ramírez Mascarúa
Issac Shakalo Paz

}

Problema y objetivo {

Problema: Anticipar ocurrencia y magnitud de asaltos por colonia y franja horaria.

Objetivos:

- Clasificación: predecir si habrá al menos un asalto (0/1).
- Regresión: estimar el conteo esperado de asaltos (número entero ≥ 0) por combinación (colonia, mes, día, turno).

Restricciones de la tarea: usar técnicas y modelos vistos en clase; dataset con ≥100 train y ≥20% test; medir Accuracy, Precision, Recall, F1, ROC/AUC; entregar reporte + interfaz de prueba.

}

Datos y etiquetas {

Fuente de datos:

Encuesta propia mediante Google Forms, difundida en 12 grupos de Facebook y 6 servidores de Discord de la Ciudad de México. Se recopilaron 217 respuestas válidas

Estructura de datos:

- colonia, mes (texto), dia_semana (texto), turno (categoría),
- asalto {Sí/No} para clasificación,
- conteo por cubeta (colonia, mes, día, turno) para regresión (obtenido con groupby sobre las respuestas).

Preguntas del formulario (ligadas a las features):

- **Colonia principal que frecuentas** (vives / trabajas / estudias): Selecciona una colonia de la CDMX.
- **Mes menos seguro en tu colonia**: Enero… Diciembre. (Se usará para mes_num y para las variables cíclicas mes_sin/mes_cos).
- **Día de la semana menos seguro:** Lunes… Domingo.
- **Turno menos seguro**: Madrugada (00:00-05:59), Mañana (06:00-11:59), Tarde (12:00-17:59), Noche (18:00-23:59).
- **Asalto**: En los últimos 12 meses, ¿viviste o presenciaste al menos un asalto en la combinación de colonia/mes/día/turno que elegiste? (Sí/No).

}

Datos y etiquetas {

}

Ingeniería de características

Mes cíclico:

mes_sin = sin(2π*mes_num/12), mes_cos = cos(2π*mes_num/12) (captura periodicidad anual).

Captura periodicidad anual (diciembre y enero son consecutivos)

Colonia:

one-hot

Transforma 46 colonias en variables binarias para análisis espacial

Turno:

Mapeo a 4 franjas {0,1,2,3} y luego one-hot.
Representa franjas horarias como variables categóricas independientes

Día de la semana:

texto → one-hot

Convierte días en 7 variables binarias para capturar patrones semanales

}

Construcción de datasets {

Clasificación (X_clf, y_clf):

- get_dummies en {colonia, dia_semana, turno} + {mes_sin, mes_cos}.
- y_clf = asalto ∈ {0,1}.

Predecir si ocurrirá al menos un asalto en la combinación dada

Regresión (X_reg, y_reg):

- groupby([colonia, mes, dia_semana, turno, mes_num, mes_sin, mes_cos]).size() → conteo.
- get_dummies y reindex para alinear columnas con las de clasificación.
- Normalización (train only) y columna bias.

Estimar la cantidad exacta de asaltos esperados

- Split: 80/20 (train/test). Para la regresión, dentro del train usamos 25% validación para buscar hiperparámetros.

}

Modelos y entrenamiento {

Clasificación — Regresión Logística (scikit-learn):

- Búsqueda de hiperparámetros con GridSearchCV (cv=5):
- C ∈ {0.01, 0.1, 1, 10}, penalty ∈ {l1, l2}, solver ∈ {liblinear, saga}.
- Evaluación: Accuracy, Precision, Recall, F1 y ROC/AUC.
- Validación adicional: cross_validate (5 folds) con los mejores hiperparámetros.

Regresión — Lineal Manual (Gradiente Descendente)

- Implementación desde cero (MSE) con búsqueda manual de alpha e iteraciones.
- Hiperparámetros probados: alpha ∈ {0.001, 0.01, 0.1, 0.5, 1}, iter ∈ {500,1000,1500,2000}.
- Métrica: MSE (train/val/test).

}

Métricas y resultados {

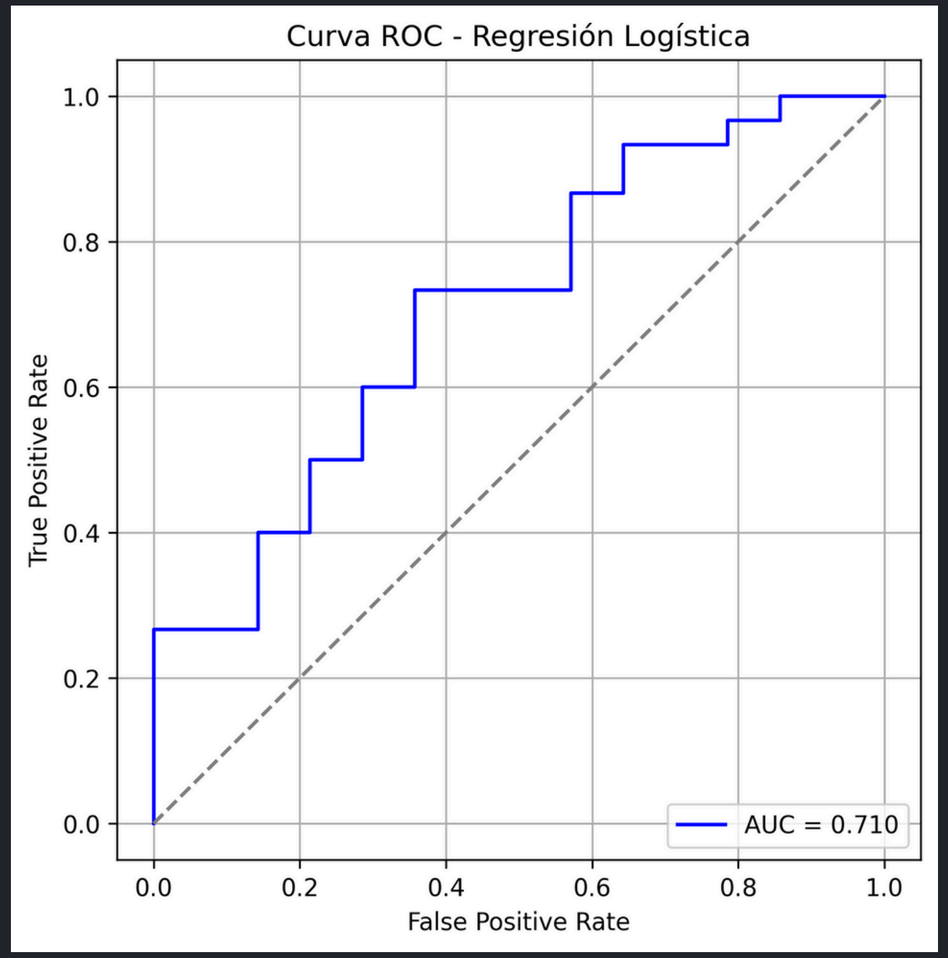
Clasificación — Regresión Logística (mejores hiperparámetros via GridSearchCV):

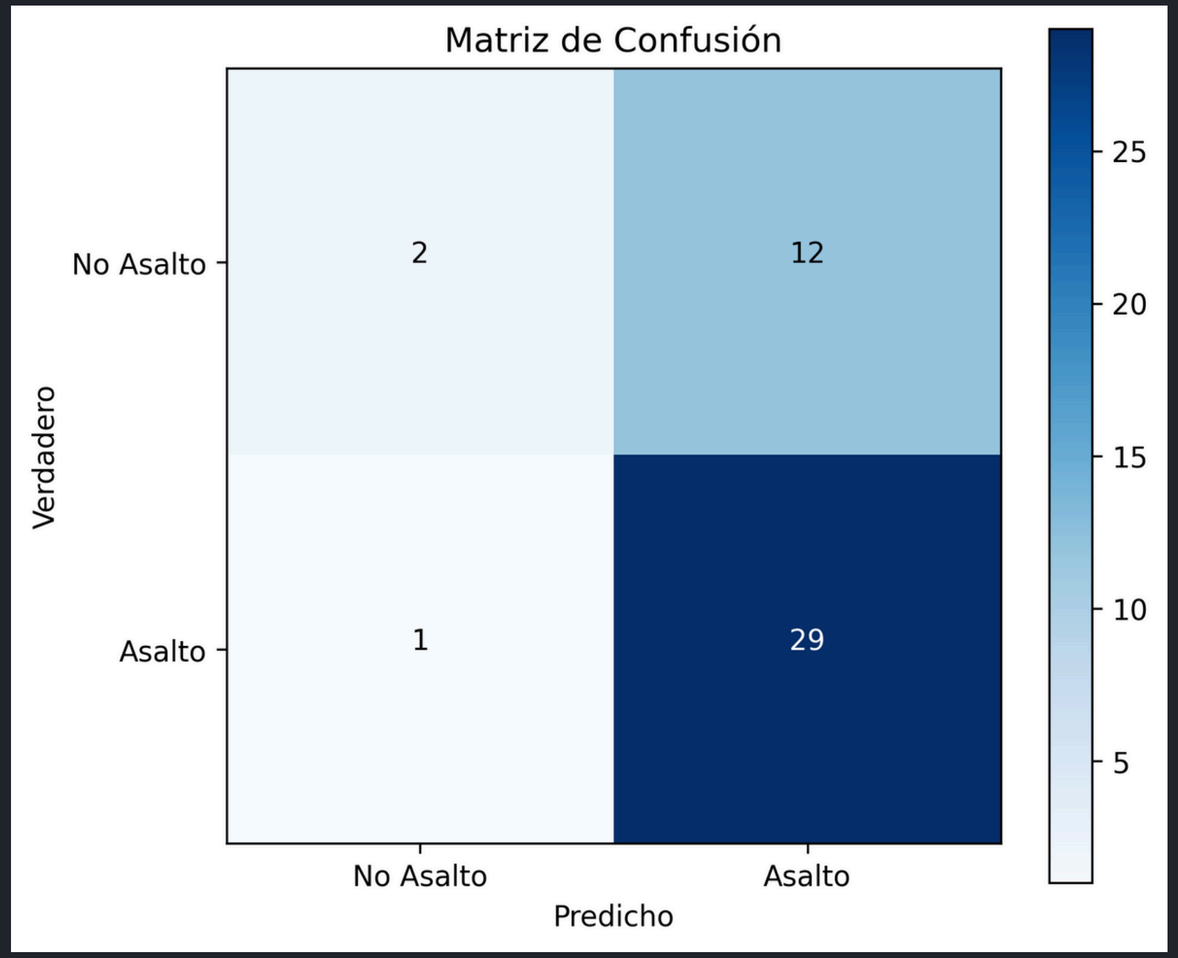
- Mejores parámetros: C=0.1, penalty=L2, solver=liblinear.
- ROC/AUC (test): 0.710 (separación moderada de clases).
- Cross-Validation (5 folds):
- Accuracy (prom): 0.802
- Precision (prom): 0.785
- Recall (prom): 0.987
- F1 (prom): 0.874
- Mejor F1 en CV (de GridSearch): 0.888

Regresión — Lineal manual (gradiente descendente):

- Búsqueda manual: mejor alpha=0.1, iteraciones=500.
- Costo validación: 0.2074
- MSE/Costo entrenamiento: 0.03379
- MSE/Costo prueba: 0.01044

}


Curva ROC - Regresión Logística
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 0.710


Matriz de Confusión
Verdadero
Predicho
No Asalto
Asalto
2
12
1
29
25
20
15
10
5

}

Interpretación y hallazgos {

- Recall alto (≈0.99): el clasificador recupera casi todos los casos positivos (pocos falsos negativos). Útil en escenarios donde perder un asalto es costoso; implica tolerar más falsos positivos (precision ≈0.78).
- AUC ≈0.71: capacidad de separación moderada; hay margen para mejorar señal (más variables o mejor balanceo de clases/negativos).

- Regularización: GridSearch eligió L2 con C=0.1, indicando que regularización fuerte ayuda a generalizar (evita sobreajuste en muchas dummies de colonia/día/turno).
- Regresión lineal: con alpha=0.1 y 500 iteraciones converge rápido; MSE test ≈0.010 sugiere buen ajuste para conteos bajos (la mayoría de cubetas tienen 0-1 eventos).
- Features temporales: el uso de mes_sin/mes_cos aporta estacionalidad; el turno y el día contribuyen a la señal.

}

Interfaz de prueba {

- **Streamlit:** formulario con (colonia, mes, día, turno).
- **Flujo:** 1) predicción logística (probabilidad y 0/1), 2) si 1, regresión lineal estima conteo.

}

Limitaciones {

- Datos faltantes / subregistro: Sesgos en reporte impactan el modelo.
- Granularidad espacial: colonia ≠ zonas exactas donde ocurren eventos.
- No causalidad: patrones ≠ causas; evitar usos punitivos o discriminatorios.
- Desbalance posible en clases → monitorear métricas por clase.

}

Próximos pasos {

- Reforzar Aspectos Éticos y de Causalidad
- Balancear Representación de Clases
- Mejorar Granularidad Espacial
- Abordar Datos Faltantes / Subregistro

}

Resumen final {

- Pipeline de datos → features cíclicas + one-hot.
- Clasificador logístico con grid search y CV.
- Regresor lineal manual con búsqueda de hiperparámetros.

}