Clasificación de sexismo en tweets

MIARFID - ALC - Laboratorio 1

Shiyi Cheng - Pablo Segovia Martínez

28 de febrero de 2026

1. Introducción

Este proyecto aborda la tarea de detección automática de contenido sexista en publicaciones de X, utilizando el dataset EXIST 2025. El objetivo es desarrollar y evaluar diferentes aproximaciones de clasificación binaria (sexista/no sexista) mediante modelos clásicos y modelos de lenguaje pre-entrenados con fine-tuning.

2. Metodología

2.1. Preprocesamiento de datos

Se han implementado dos versiones de preprocesamiento con mejoras progresivas:

2.1.1. Versión 1 (V1)

- **tweet original**: texto sin modificaciones
- **text clean**: limpieza básica (lowercase, elimina URLs/menciones/hashtags)
- Sin eliminación de stopwords
- Sin manejo de negaciones
- Mantiene acentos españoles
- 13 features de ingeniería

2.1.2. Versión 2 (V2) - Mejoras implementadas

- **tweet original**: texto sin modificaciones
- **text clean mejorado**:
  - **eliminación de stopwords**: 313 stopwords del corpus NLTK español
  - **preservación de negaciones**: mantiene {no, nunca, jamás, nada, nadie, tampoco, ni, sin}
  - **marcado de contexto negativo**: palabras tras negaciones se marcan con NEG_
  - **normalización de acentos**: á→a, é→e, ñ→n
  - **normalización de elongaciones**: "siiii"→ "sii"

- **detección de emojis mejorada**: biblioteca `emoji` (más precisa)

- 16 features de ingeniería (+3 nuevas): n_caps_words, n_elongations, n_negations
- **reducción de texto**: ∼52 % menos palabras en text_clean vs V1

Impacto clave de V2: La preservación y marcado de negaciones es **crítica** para detección de sexismo, ya que la negación cambia completamente el significado semántico del texto.

2.2. Modelos evaluados

Se ha experimentado con tres familias de modelos:

2.2.1. Modelos clásicos

Modelos de machine learning tradicionales utilizando vectorización TF-IDF y Bag-of-Words:

- Regresión Logística
- Support Vector Machines (LinearSVC, SVC-RBF)
- Árboles de Decisión y Random Forest
- Gradient Boosting y AdaBoost
- Naive Bayes (Multinomial, Complement, Bernoulli)
- K-Nearest Neighbors
- Ensambles (Voting, Bagging, Stacking)

2.2.2. Modelos de lenguaje Fine-tuned

Modelos transformer pre-entrenados ajustados con LoRA (Low-Rank Adaptation):

- **F2LLM-4B**: Modelo de 4B parámetros especializado en español
- **KaLM**: Modelo multilingüe con soporte para español

2.2.3. Modelos generativos

Modelos grandes de lenguaje evaluados en modo zero-shot e inference con fine-tuning:

- **Ministral-3-8B-Instruct**: Modelo instruccional de 8B parámetros con cuantización FP8

3. Resultados

3.1. Evolución entre versiones

Se realizaron dos iteraciones completas del proyecto (V1 y V2), con mejoras sustanciales en el preprocesamiento (eliminación de stopwords, preservación de negaciones, normalización de acentos) y ajustes en los hiperparámetros de entrenamiento. **Nota importante**: Las mejoras del preprocesamiento solo afectan la variante text_clean; la variante tweet utiliza texto idéntico en ambas versiones.

Modelo	Texto	Ver	Acc	Prec	Rec	F1
F2LLM-4B	tweet	V1	0.8604	0.8294	0.8642	0.8464
	tweet	V2	**0.8593**	0.7966	**0.9185**	**0.8532**
	clean	V1	0.8473	0.8122	0.8543	0.8327
	clean	V2	0.8341	0.7581	**0.9210**	0.8317
KaLM	tweet	V1	0.8363	0.8137	0.8198	**0.8167**
	tweet	V2	0.8143	0.7682	0.8346	0.8000
	clean	V1	0.8363	0.7963	**0.8494**	**0.8220**
	clean	V2	—	—	—	—
Ministral-3-8B	ZS	V1	0.8264	0.7892	0.8321	**0.8101**
	ZS	V2	0.8143	0.7500	**0.8741**	0.8073
	FT	V1	**0.8451**	**0.8587**	0.7802	**0.8176**
	FT	V2	0.5725	0.9000	0.0444	0.0847

Tabla 1: Comparativa Completa V1 vs V2

3.1.1. Comparativa Completa: Modelos LLM

La Tabla 1 muestra la evolución del rendimiento en todos los modelos de lenguaje con sus variantes de preprocesamiento.

Observaciones clave:

- **F2LLM-4B (tweet) V2**: Mejora significativa en recall (+5.43 pp) y F1 (+0.68 pp). Tienen hiperparámetros idénticos, texto tweet idéntico, pero threshold óptimo muy diferente y AUC casi igual. La mejora se debe a **varianza aleatoria** (inicialización LoRA, shuffle), NO preprocesamiento ni hiperparámetros.
- **F2LLM-4B (clean) V2**: Alcanza el recall más alto (0.9210) sacrificando precisión. El texto clean V2 reduce ruido manteniendo contexto negativo.
- **KaLM V2**: Ligero empeoramiento contra V1. Posible overfitting al preprocesamiento V1 o incompatibilidad con eliminación agresiva de stopwords.
- **Ministral-3-8B FT V2**: El fine-tuning ha fallado drásticamente en comparación con la V1.

3.1.2. Comparativa: Modelos Clásicos

La Tabla 2 muestra la evolución de los mejores modelos clásicos.

Modelo	Ver	Acc	Prec	Rec	F1
Stacking	V1	**0.7846**	**0.7895**	**0.7037**	**0.7441**
Stacking	V2	0.7824	0.7867	0.7012	0.7415
LogReg + TF-IDF	V1	0.7637	0.7699	0.6691	0.7160
LogReg + TF-IDF	V2	**0.7725**	**0.7845**	**0.6741**	**0.7251**

Tabla 2: Comparativa V1 vs V2 - Modelos clásicos

Evolución en V2 (modelos clásicos):

- Stacking: -0.26 pp en F1, -0.25 pp en recall (ligero empeoramiento)

- LogReg: +0.91 pp en F1, +0.50 pp en recall (mejora modesta)
- La eliminación de stopwords (313 palabras) tiene impacto mixto: beneficia LogReg pero perjudica ligeramente Stacking

3.2. Métricas Finales en Conjunto de Validación (V2)

La Tabla 3 muestra los resultados finales de los modelos de lenguaje en V2, mientras que la Tabla 4 presenta los mejores modelos clásicos.

Modelo	Accuracy	Precision	Recall	F1-Score
F2LLM-4B (tweet)	**0.8593**	0.7966	**0.9185**	**0.8532**
F2LLM-4B (clean)	0.8341	0.7581	0.9210	0.8317
KaLM (tweet)	0.8143	0.7682	0.8346	0.8000
Ministral-3-8B (ZS)	0.8143	0.7500	0.8741	0.8073
Ministral-3-8B (FT)	0.5725	0.9000	0.0444	0.0847

Tabla 3: Resultados de Modelos de Lenguaje V2 (DEV)

Modelo	Accuracy	Precision	Recall	F1-Score
Stacking	**0.7824**	0.7867	0.7012	**0.7415**
Logistic Regression	0.7725	**0.7845**	0.6741	0.7251
Gradient Boosting	0.7736	0.8373	0.6099	0.7057
LinearSVC	0.7736	0.7901	0.6691	0.7246
Bagging (LR)	0.7703	0.7849	0.6667	0.7210

Tabla 4: Mejores Modelos Clásicos V2 (DEV) - TF-IDF

3.3. Análisis Comparativo

3.3.1. Impacto del Preprocesamiento: Tweet vs Clean

La Tabla 5 compara el rendimiento de los textos originales (tweet) versus los preprocesados (text_clean) en ambas versiones.

Modelo	Versión	Tweet F1	Clean F1	Mejor	ΔF1
F2LLM-4B	V1	0.8464	0.8327	Tweet	-0.0137
	V2	0.8532	0.8317	Tweet	-0.0215
KaLM	V1	0.8167	0.8220	Clean	+0.0053
	V2	0.8000	—	Tweet	—

Tabla 5: Impacto del Preprocesamiento en F1-Score. ΔF1 = Clean - Tweet.

Conclusiones sobre el preprocesamiento:

- **F2LLM-4B**: Los textos originales (tweet) superan consistentemente a los preprocesados en ambas versiones. La diferencia se amplía en V2 (-0.0215 vs -0.0137 en V1), principalmente porque text_clean V2 empeoró ligeramente (-0.10 pp) al eliminar más información (stopwords), mientras tweet V2 mejoró por ajustes de entrenamiento (+0.68 pp). Esto

confirma que el texto original preserva información útil (URLs, menciones, emojis) para detección de sexismo.

- **KaLM V1**: Único caso donde text_clean supera ligeramente a tweet (+0.0053). La limpieza agresiva puede beneficiar modelos más pequeños reduciendo ruido.
- **Limpieza V2 vs V1**: A pesar de las mejoras en V2 (preservación de negaciones, normalización de acentos), el text_clean sigue siendo inferior al texto original para modelos grandes. Esto indica que los transformers pre-entrenados ya manejan bien el ruido y se benefician del contexto completo.
- **Recomendación general**: Para modelos LLM, usar texto original (tweet). Para modelos clásicos (TF-IDF), el preprocesamiento V2 mejora significativamente el rendimiento.

3.3.2. Evolución V1 → V2: Análisis Detallado

La Tabla 6 cuantifica las mejoras/empeoramientos entre versiones.

Modelo (Tweet)	ΔAcc	ΔPrec	ΔRec	ΔF1
F2LLM-4B	-0.0011	-0.0328	+0.0543	+0.0068
KaLM	-0.0220	-0.0455	+0.0148	-0.0167
Ministral-3B (ZS)	-0.0121	-0.0392	+0.0420	-0.0028
Stacking	-0.0022	-0.0028	-0.0025	-0.0026
LogReg (TF-IDF)	+0.0088	+0.0146	+0.0050	+0.0091

Tabla 6: Cambios entre V1 y V2 (Δ = V2 - V1). En azul mejoras >+0.01, en rojo empeoramientos <-0.01.

Análisis de cambios:

- **F2LLM-4B (tweet)**: Mejora sustancial en recall (+5.43 pp) y F1 (+0.68 pp). **Importante**: Como el texto tweet es idéntico entre versiones e hiperparámetros iguales (verificado en notebooks), esta diferencia proviene de **varianza aleatoria del entrenamiento** (inicialización de pesos LoRA, shuffle, semillas). Threshold óptimo divergió significativamente (0.3812→0.2214), favoreciendo recall en V2. Trade-off: -3.28 pp en precisión.
- **KaLM**: Empeora en todas las métricas. Hipótesis: (1) El modelo fue pre-entrenado con texto menos procesado, (2) la eliminación agresiva de stopwords elimina patrones importantes, (3) KaLM requiere ajustes de hiperparámetros para V2.
- **Modelos clásicos**: Impacto mixto en V2. LogReg mejora ligeramente (+0.91 pp F1), pero Stacking empeora (-0.26 pp F1). La eliminación de 313 stopwords beneficia modelos simples pero puede perjudicar ensambles.
- **Ministral-3B ZS**: Ligero empeoramiento en F1 (-0.0028), pero mejora recall (+4.20 pp). Trade-off precision-recall ajustado por preprocesamiento.

3.3.3. Rendimiento por Familia de Modelos

- **Modelos de Lenguaje (V2)**: Los modelos transformer fine-tuned superan significativamente a los modelos clásicos, con mejoras de hasta **+10.93 puntos** en F1-Score (F2LLM-4B: 0.8532 vs Stacking: 0.7415). El modelo F2LLM-4B alcanza el mayor recall (0.9185) y mejor F1, demostrando superioridad en comprensión contextual para detección de sexismo.

- **Modelos Clásicos (V2)**: El ensamble mediante Stacking alcanza el mejor rendimiento entre modelos clásicos V2 (F1=0.7415), superando a la regresión logística simple en +1.64 puntos porcentuales (F1=0.7251). Sin embargo, Stacking V1 obtuvo F1=0.7441, ligeramente superior. Son computacionalmente más eficientes pero menos precisos que LLMs.
- **Impacto del Preprocesamiento en V2**:
  - **LLMs**: Los textos originales (tweet) superan consistentemente a los preprocesados. F2LLM-4B: tweet (0.8532) vs clean (0.8317), Δ=-0.0215. Los transformers aprovechan mejor el contexto completo (URLs, emojis, menciones) que el preprocesamiento elimina.
  - **Modelos clásicos**: El preprocesamiento V2 tiene impacto mixto. LogReg V2: +0.91 pp F1 vs V1 (mejora modesta). Stacking empeora -0.26 pp F1. La eliminación de 313 stopwords beneficia modelos simples pero puede perjudicar ensambles.
  - **preservación de negaciones crítica**: V2 mantiene {no, nunca, jamás, nada, nadie, tampoco, ni, sin} y marca contexto con NEG_ en text_clean, mejorando recall en F2LLM-4B (clean): 0.9210 vs 0.8543 en V1 (+6.67 pp). **No afecta variante tweet** (texto idéntico).
- **Ensemble de Top 5 Modelos (V2)**: Se evaluó un ensamble por votación mayoritaria combinando F2LLM-4B (tweet y clean), KaLM (tweet), Ministral-3B y Regresión Logística. El resultado obtuvo **F1=0.8532, Acc=0.8593, Recall=0.9185**, **idéntico al mejor modelo individual** (F2LLM-4B tweet). Esto indica:
  - El modelo F2LLM-4B domina las predicciones del ensemble (90 %+ de votos)
  - Los demás modelos no aportan suficiente diversidad o precisión
  - La votación mayoritaria no corrige errores del mejor modelo

  Conclusión sobre ensemble: No aporta mejora sobre F2LLM-4B individual, sugiriendo que este modelo ya alcanza el rendimiento óptimo con los datos disponibles.

3.3.4. Mejores Configuraciones (V2)

1. **Mayor F1-Score**: F2LLM-4B con tweet original **(0.8532)** — mejor modelo general
2. **Mayor Recall**: F2LLM-4B con text_clean **(0.9210)** — maximiza detección de casos positivos (alternativa: tweet con 0.9185)
3. **Mayor Precision**: Ministral-3-8B FT (versión fallida excluida); entre modelos viables: Gradient Boosting **(0.8373)**
4. **Mayor Accuracy**: F2LLM-4B con tweet original **(0.8593)** — equilibrio global óptimo
5. **Mejor modelo clásico V2**: Stacking con TF-IDF **(F1: 0.7415)** — nota: V1 alcanzó F1=0.7441, ligeramente superior
6. **Ensemble Top 5**: Votación mayoritaria **(F1: 0.8532)** — no mejora sobre F2LLM-4B individual

Recomendación final: Para la competición EXIST 2025, el modelo **F2LLM-4B con texto original (V2)** es la mejor opción:

- Mejor F1-Score general: 0.8532

- Excelente recall: 0.9185 (detecta 91.85 % de casos sexistas)
- El ensemble no aporta mejora adicional
- Balance óptimo entre rendimiento y complejidad

4. Conclusiones

1. **Superioridad de LLMs sobre modelos clásicos**: Los modelos de lenguaje pre-entrenados con fine-tuning superan ampliamente a los métodos clásicos de ML, con **mejoras de hasta +10.93 puntos** en F1-Score (F2LLM-4B: 0.8532 vs Stacking: 0.7415). Esta ventaja se explica por su capacidad de comprensión contextual profunda, esencial para detectar sexismo implícito o irónico.

2. **Modelo óptimo: F2LLM-4B (tweet) V2**: Alcanza el mejor rendimiento global (**F1=0.8532, Recall=0.9185, Accuracy=0.8593**), mejorando +0.68 pp F1 y +5.43 pp recall sobre V1. El fine-tuning con LoRA sobre tweet original (sin preprocesamiento) maximiza la capacidad de detección. **Análisis de notebooks revela**: Hiperparámetros idénticos entre V1/V2 (LR=5e-5, epochs=5, LoRA r=16), tweet idéntico, pero thresholds óptimos muy diferentes (0.3812 vs 0.2214). AUC prácticamente igual (0.9303 vs 0.9307). **Conclusión**: La mejora es producto de **varianza aleatoria** del entrenamiento (inicialización de pesos, shuffle de batches), no de cambios sistemáticos. V2 convergió a mínimo local que favorece recall.

3. **Impacto crítico del preprocesamiento V2**: Las mejoras implementadas tienen efectos diferenciados:

   - **preservación de negaciones + marcado NEG_**: cambio más importante del preprocesamiento. Beneficia detección en text_clean donde negaciones son semánticamente críticas. **Nota**: La mejora de F2LLM-4B (tweet) +5.43 pp recall NO proviene de esto, ya que tweet es idéntico en V1/V2.
   - **eliminación de 313 stopwords españolas**: reduce vocabulario 52 % con impacto mixto en modelos clásicos (+0.91 pp LogReg, -0.26 pp Stacking). Solo afecta variante text_clean.
   - **normalización de acentos**: reduce variabilidad léxica sin pérdida de información (solo text_clean).

4. **Texto original superior para LLMs**: Contra la intuición inicial, los textos sin preprocesar (tweet) superan consistentemente a los limpios (text_clean) en modelos transformer. F2LLM-4B: 0.8532 (tweet) vs 0.8317 (clean), Δ=-2.15 pp. Los transformers pre-entrenados ya manejan ruido eficientemente y se benefician del contexto completo (URLs, emojis, menciones).

5. **Ensemble sin mejora**: El ensamble por votación mayoritaria (Top 5 modelos) alcanza F1=0.8532, **idéntico** al mejor individual. Esto sugiere:

   - F2LLM-4B domina predicciones (90 %+ votos coincidentes)
   - Falta diversidad en arquitecturas/estrategias
   - El mejor modelo ya maximiza rendimiento con datos disponibles

   Recomendación: Usar F2LLM-4B individual (simplicidad, eficiencia, mismo resultado).

6. **Fine-tuning con LoRA: éxitos y fracasos**:
   - **éxito**: F2LLM-4B y KaLM ajustados eficientemente (recursos limitados) con resultados competitivos.
   - **fracaso**: Ministral-3-8B FT colapsó totalmente (F1: 0.0847, Recall: 0.0444). Hipótesis: LR inadecuada, datos corruptos, o incompatibilidad LoRA-FP8. Requiere **supervisión cuidadosa** del proceso.
7. **Evolución V1 → V2 muestra rol de varianza aleatoria**: La segunda iteración logró mejoras en F2LLM-4B (tweet) por **varianza aleatoria del entrenamiento** (no cambios sistemáticos), y mejoras modestas reales en modelos clásicos con preprocesamiento mejorado (LogReg +0.91 pp F1). Stacking empeoró (-0.26 pp F1), demostrando que el preprocesamiento agresivo no siempre beneficia ensambles. Lección: **Fijar semillas aleatorias** y ejecutar múltiples entrenamientos para distinguir mejoras reales de varianza estocástica.
8. **Trade-off Precision-Recall**: V2 prioriza recall sobre precision:
   - F2LLM-4B V2: Recall 0.9185 (+5.43 pp) a costa de Precision 0.7966 (-3.28 pp)
   - Justificado en detección de sexismo: **mejor detectar más casos (recall) aunque genere algunos falsos positivos**, que perder casos reales (recall bajo).

4.1. Trabajo Futuro

- **Implementar reproducibilidad experimental**: El análisis reveló que diferencias entre V1/V2 en F2LLM-4B (tweet) son por varianza aleatoria, no mejoras sistemáticas. Para trabajo futuro:
  - Fijar semillas aleatorias: `torch.manual_seed()`, `np.random.seed()`, `random.seed()`
  - Configurar `deterministic=True` en PyTorch cuando sea posible
  - Ejecutar **múltiples entrenamientos (3-5 runs)** con diferentes semillas
  - Reportar media ± desviación estándar en métricas clave
  - Aplicar tests estadísticos (t-test, Wilcoxon) para validar significancia de mejoras

  Esto permitirá distinguir mejoras reales de fluctuaciones estocásticas del proceso de entrenamiento.
- **Investigar el fallo de Ministral-3-8B fine-tuning**: Analizar por qué el recall colapsó a 0.0444 y F1 a 0.0847. Plantear experimentos controlados:
  - Probar diferentes learning rates: [1e-5, 5e-5, 1e-4]
  - Ajustar warmup steps y schedule (linear, cosine)
  - Verificar configuración LoRA (rank, alpha, dropout)
  - Detectar posible corrupción en datos de entrenamiento
  - Evaluar compatibilidad LoRA con cuantización FP8
- **Análisis de errores cualitativo**: Identificar patrones lingüísticos específicos donde los modelos fallan sistemáticamente:
  - Casos de ironía y sarcasmo (muy comunes en redes sociales)
  - Referencias culturales o memes (contexto externo necesario)

  - Sexismo implícito vs explícito (diferencias en detección)
  - Falsos positivos recurrentes (crítica legítima vs sexismo)

- **Explorar ensambles avanzados**: Dado que votación simple no mejora, probar estrategias sofisticadas:
  - **Stacking con meta-aprendizaje**: Entrenar meta-modelo (LogReg, XGBoost) sobre predicciones de LLMs
  - **Ponderación basada en confianza**: Asignar pesos dinámicos según softmax scores
  - **Ensambles especializados**: Combinar modelos complementarios (uno con alto recall, otro con alta precision)
  - **Error-correcting ensembles**: Entrenar modelos secundarios específicamente en errores del primario
- **Optimización exhaustiva de hiperparámetros**: Búsqueda sistemática más allá de ajustes manuales:
  - Grid/Random Search para LoRA: rank [4, 8, 16, 32], alpha [8, 16, 32, 64]
  - Learning rate: [1e-5 a 1e-3] con diferentes schedules
  - Batch size y gradient accumulation steps
  - Dropout en capas LoRA [0.0, 0.05, 0.1]
  - Evaluar impacto de cada hiperparámetro con Optuna o Ray Tune
- **Evaluar modelos más recientes**: Probar arquitecturas state-of-the-art y específicas de español:
  - **Llama 3**: 8B/70B con instrucciones en español
  - **Mixtral 8x7B**: Mixture-of-Experts, mejor capacidad
  - **RoBERTa-es / BETO**: Transformers entrenados exclusivamente en español
  - **MarIA / BERTIN**: Modelos GPT/BERT españoles del CLARIN
  - **mBERT / XLM-RoBERTa**: Multilingües con fuerte representación de español
- **Augmentación de datos**: Generar ejemplos sintéticos para balancear clases y mejorar generalización:
  - **Back-translation**: Traducir tweets a inglés y volver a español (paráfrasis)
  - **Parafraseo con LLMs**: Usar GPT-4/Mixtral para generar variaciones semánticas
  - **Synonym replacement**: Sustituir palabras por sinónimos (WordNet español)
  - **Synthetic data generation**: Generar tweets sexistas/no-sexistas con prompts específicos
  - Validar calidad de datos sintéticos con anotadores humanos
- **Análisis de sesgo y equidad**: Evaluar comportamiento del modelo en diferentes subgrupos:
  - Rendimiento por tipo de sexismo (explícito, implícito, benevolente, hostil)
  - Diferencias en detección según género del objetivo
  - Evaluar falsos positivos que censuran crítica legítima al patriarcado

  - Análisis de fairness metrics (equalized odds, demographic parity)

- **Explicabilidad e interpretabilidad**: Comprender decisiones del modelo:
  - **LIME/SHAP**: Identificar palabras/frases más influyentes por predicción
  - **Attention visualization**: Mapas de atención para interpretar contexto relevante
  - **Probing classifiers**: Evaluar qué información captura cada capa
  - Generar rationales automáticos (explicar por qué es sexista)
- **Explorar preprocesamiento híbrido**: Dado que tweet original funciona mejor en LLMs pero clean en clásicos:
  - Preprocesamiento selectivo según modelo
  - **Feature injection**: Concatenar features de text_clean como tokens especiales en tweet original
  - Evaluar impacto de preservar solo emojis o solo menciones