Proyecto Machine Learning – Movimiento ocular

Jasmin Juliana Jaramillo Tapias, Andrés Darío Higuita Pérez y Juan Guillermo Preciado Zapata.

***Resumen*— El presente trabajo aborda el estudio y análisis de diversos modelos de aprendizaje automático aplicados a problemas de predicción y clasificación. Estos enfoques son de vital importancia, ya que permiten transformar datos en conocimiento útil, adquiriendo una relevancia cada vez mayor en la toma de decisiones basadas en información.**

A lo largo de las etapas de exploración, entrenamiento y validación de modelos, se demuestra cómo las técnicas de machine learning son capaces de identificar patrones complejos y generar respuestas precisas en diversos contextos de la vida cotidiana, científica e industrial.

Dentro de este marco, se desarrolla un caso de aplicación práctico utilizando la base de datos "Eye Movements", orientado al análisis de patrones visuales y su relación con la relevancia de la información leída. Finalmente, se realiza un análisis comparativo de los métodos empleados, evaluando su capacidad para generalizar y adaptarse a distintos tipos de datos y escenarios. El trabajo reflexiona sobre el papel del aprendizaje automático como una herramienta fundamental para la innovación tecnológica y la comprensión de fenómenos complejos.

***Índice de términos*— Machine Learning, Predicción, Clasificación, Validación, Inteligencia Artificial, Eye Movements.**

I. INTRODUCCIÓN

En los últimos años, el análisis de movimientos oculares (eye tracking) ha adquirido gran relevancia en el estudio del comportamiento humano, la atención visual y los procesos cognitivos durante la lectura o navegación en entornos digitales. La información derivada de los movimientos oculares permite inferir qué partes de un texto o imagen captaron la atención del observador, así como medir el nivel de comprensión, interés o relevancia percibida. El presente trabajo aborda el problema de clasificar la relevancia de oraciones leídas a partir de los movimientos oculares registrados durante una tarea de lectura, utilizando la base de datos Eye Movements disponible en OpenML. Este problema se enmarca en la categoría de aprendizaje supervisado, donde el objetivo es predecir una etiqueta de clase asociada a cada palabra leída: irrelevante, relevante o correcta. El uso de técnicas de Machine Learning en este contexto resulta especialmente valioso, ya que permite automatizar el análisis de señales complejas y multidimensionales, capturadas mediante dispositivos de eye-tracking, con el fin de construir modelos capaces de inferir patrones cognitivos y conductuales. El desarrollo de este tipo de soluciones tiene aplicaciones en campos como la evaluación de interfaces, la neurociencia cognitiva, la educación y la publicidad digital.

II. DESCRIPCIÓN DEL PROBLEMA

El problema planteado consiste en clasificar palabras leídas según su nivel de **relevancia** para una pregunta dada, basándose en las características obtenidas de los movimientos oculares de distintos sujetos durante la lectura. Una solución basada en **aprendizaje automático** es adecuada porque permite modelar relaciones complejas entre las variables (características oculares) y los niveles de relevancia de las palabras leídas, automatizando la clasificación con mayor precisión y eficiencia que los métodos analíticos comunes.

El conjunto de datos **Eye Movements** contiene instancias que representan palabras individuales observadas en múltiples tareas de lectura. Cada tarea o "asignación" consiste en una pregunta y diez oraciones (títulos de noticias). De estas oraciones, una corresponde a la respuesta correcta (Correcta), cuatro son Relevantes, pero no responden directamente a la pregunta, y cinco son Irrelevantes.

El conjunto de datos está identificado en OpenML con el ID = 1044, está conformado por 10936 registros y 28 variables que describen el comportamiento ocular de distintos participantes durante tareas de lectura. La primera columna (lineNo) identifica la línea del texto, la segunda (assgNo) corresponde al número de asignación o tarea experimental, y las columnas 3 a 24 contienen las características principales de los movimientos oculares. Dentro de este grupo se incluyen variables numéricas –que cuantifican aspectos temporales, espaciales y fisiológicos, como duraciones de fijaciones, longitudes de sacadas y variaciones pupilares– y variables categóricas, que representan estados o posiciones específicas durante el proceso de lectura, como las ubicaciones de fijación o la presencia de regresiones.

Las columnas 25 a 27 recogen información contextual y estructural del texto, como el número total de palabras en la oración, la posición del título y la ubicación ordinal de cada palabra. Finalmente, la columna 28 (label) corresponde a la variable objetivo del modelo supervisado, con tres categorías de clasificación: 0 = Irrelevante (Distribución del 34,78%), 1 = Relevante (Distribución del 38,97%) y 2 = Correcta (Distribución del 24,24%) con distribuciones que se puede visualizar en la **Fig. 1**. Las características y su descripción se resumen en la **TABLA I**.

El conjunto de datos se encuentra preprocesado y estructurado en formato **ARFF** (Attribute-Relation File Format), compatible con la herramienta Weka y ampliamente utilizado en entornos de aprendizaje automático. Este formato permite representar de manera explícita los nombres, tipos y valores de cada atributo, así como la variable objetivo de clasificación, lo que garantiza una correcta interpretación por parte de

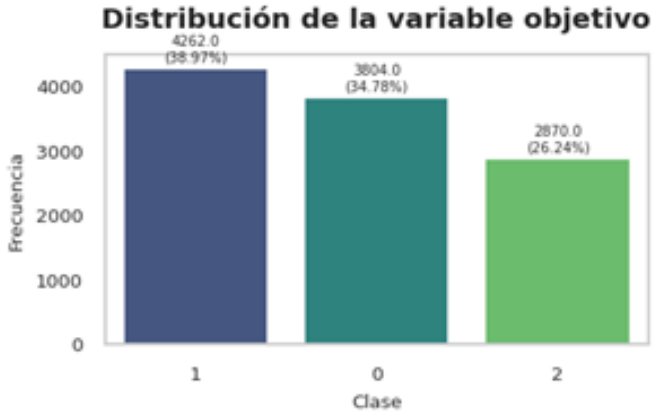

Fig. 1. Distribución de cada clasificación.

diferentes librerías y entornos de análisis, para nuestro caso Python.

El conjunto de datos no presenta valores faltantes en ninguna de sus variables, lo que facilita su utilización directa en los procesos de análisis y modelado sin requerir estrategias de imputación o limpieza adicional. Dado que el objetivo es predecir una categoría discreta, el problema se enmarca en una clasificación supervisada multiclase, donde se busca representar el nivel de relevancia de las palabras leídas. El modelo debe aprender a relacionar las características de los movimientos oculares con las clases objetivo (irrelevante, relevante y correcta).

Este paradigma resulta apropiado porque permite capturar relaciones no lineales y de alta dimensionalidad entre las variables, ofreciendo mayor capacidad predictiva y de generalización frente a los métodos tradicionales. Entre los algoritmos adecuados para abordar este tipo de problema se incluyen **Support Vector Machines (SVM), Random Forests, Redes Neuronales y Gradient Boosting.** Nota: Los valores en las columnas marcadas con un asterisco (*) son los mismos para todas las apariciones de la palabra.

III. ESTADO DEL ARTE

A. Artículo 1

Saloj¨arvi, J., Puolam¨aki, K., Simola, J., Kovanen, L., Kojo, I., And Kaski, S. (2005). “Inferring Relevance from Eye Movements: Feature Extraction.” Helsinki University of Technology.

Según lo indicado en [1] este estudio constituye la fuente original de la base de datos *Eye Movements*. Los autores desarrollaron un experimento en el que los participantes respondieron preguntas mediante la lectura de títulos de noticias, mientras sus movimientos oculares eran registrados. Se propuso un enfoque de clasificación supervisada basado en la extracción de 22 características derivadas de los registros de fijación y sacadas. Para la construcción del modelo, se evaluaron varios clasificadores, entre ellos *Naive Bayes, Decision Trees y k- Nearest Neighbors (k-NN)*. La validación se realizó mediante validación cruzada estratificada, y las métricas empleadas incluyeron exactitud (*accuracy*) y tasa

TABLA I
DESCRIPCIÓN DE LAS VARIABLES DEL CONJUNTO DE DATOS *Eye Movements*

Nº	Variable	Tipo de dato	Descripción
1	lineNo	Numérica (int64)	Número de línea del texto dentro de la tarea de lectura.
2	assgNo	Numérica (int64)	Identificador de la asignación o tarea experimental (pregunta con 10 oraciones).
3	fixcount	Numérica (uint8)	Número total de fijaciones realizadas en la palabra.
4*	firstPassCnt	Numérica (uint8)	Número de fijaciones durante la primera pasada de lectura (antes de cualquier regresión).
5*	P1stFixation	Categórica	Posición de la primera fijación en la palabra (inicio, medio o final).
6*	P2stFixation	Categórica	Posición de la segunda fijación en la palabra (si existió).
7*	prevFixDur	Numérica (int64)	Duración de la fijación anterior a la palabra actual (ms).
8*	firstfixDur	Numérica (int64)	Duración de la primera fijación en la palabra actual (ms).
9*	firstPassFixDur	Numérica (int64)	Duración total de las fijaciones durante la primera pasada de lectura (ms).
10*	nextFixDur	Numérica (int64)	Duración de la fijación inmediatamente posterior (en la siguiente palabra).
11	firstSaccLen	Numérica (float64)	Longitud de la primera sacada hacia la palabra actual (en grados visuales o píxeles).
12	lastSaccLen	Numérica (float64)	Longitud de la última sacada desde la palabra actual hacia otra palabra.
13	prevFixPos	Numérica (float64)	Posición horizontal de la fijación anterior (en píxeles o proporción del ancho del texto).
14	landingPos	Numérica (float64)	Posición de aterrizaje de la mirada en la palabra actual (proporción respecto a su longitud).
15	leavingPos	Numérica (float64)	Posición desde la que el ojo abandona la palabra (píxeles o porcentaje).
16	totalFixDur	Numérica (int64)	Duración total de todas las fijaciones realizadas en la palabra (ms).
17	meanFixDur	Numérica (float64)	Duración promedio de las fijaciones sobre la palabra (ms).
18*	nRegressFrom	Numérica (uint8)	Número de regresiones iniciadas desde la palabra.
19*	regressLen	Numérica (int64)	Longitud promedio de las regresiones (en píxeles o grados visuales).
20*	nextWordRegr	Categórica	Indica si la siguiente palabra fue objeto de una regresión (sí/no).
21*	regressDur	Numérica (int64)	Duración total de las fijaciones durante una regresión desde esta palabra (ms).
22	pupilDiamMax	Numérica (float64)	Diámetro máximo de la pupila durante la lectura de la palabra (indicador de carga cognitiva).
23	pupilDiamLag	Numérica (float64)	Diferencia de diámetro pupilar respecto a la fijación anterior.
24	timePrtctg	Numérica (float64)	Proporción de tiempo dedicada a esta palabra respecto al total de la oración.
25	nWordsInTitle	Numérica (uint8)	Número total de palabras en la oración o título leído.
26	titleNo	Numérica (uint8)	Identificador de la oración o título dentro de la tarea (1–10).
27	wordNo	Numérica (uint8)	Posición ordinal de la palabra dentro de la oración.
28	label	Categórica	Variable objetivo: 0 = irrelevante, 1 = relevante, 2 = correcta.

de error promedio. Los mejores resultados se obtuvieron con *Naive Bayes*, con una precisión promedio del 73%.

B. Artículo 2

Papoutsaki, A., Gokaslan, A., Laskey, J., Huang, J., And Hays, J. (2016). "WebGazer: Scalable Webcam Eye Tracking Using User Interactions." IJCAI.

Segúnn lo indicado en [2] aunque no utiliza directamente la misma base de datos, este trabajo propone un enfoque de aprendizaje automático para inferir la atención visual a partir de movimientos oculares capturados por cámara web, demostrando la aplicabilidad del eye tracking para tareas de inferencia cognitiva.

El modelo utiliza un paradigma de aprendizaje supervisado con regresión lineal y redes neuronales para predecir la posición de la mirada. La validación se llevó a cabo con conjuntos de entrenamiento y prueba independientes, empleando métricas como el error medio cuadrático (MSE) y el error angular. Los autores lograron una precisión promedio de 1,06° de error angular, mostrando que el análisis de patrones oculares puede ser abordado eficientemente mediante modelos de machine learning livianos y escalables.

C. Artículo 3

Bhattacharya, N., Rakshit, S., Gwizdka, J., And Kogut, P. (2020). "Relevance Prediction from Eye-movements Using Semi-interpretable Convolutional Neural Networks." arXiv preprint arXiv:2001.05152.

Segúnn lo indicado en [3] en este trabajo, los autores investigan la predicción de relevancia utilizando movimientos oculares mediante un enfoque basado en redes neuronales convolucionales (CNN). A diferencia de modelos tradicionales, este artículo introduce un método para transformar los scanpaths —la secuencia espacial de fijaciones durante la lectura— en imágenes bidimensionales que luego son procesadas por una CNN diseñada específicamente para capturar patrones temporales y espaciales de la mirada.

El objetivo del estudio fue clasificar documentos según su relevancia percibida por los usuarios, sin necesidad de acceder al contenido textual, lo que convierte al enfoque en "content-independent". Para la evaluación, se emplearon métricas de exactitud y F1-score, usando validación cruzada. Los resultados indican que el modelo alcanzó una precisión superior al 80%, superando métodos clásicos de machine learning basados únicamente en características estadísticas de fijaciones y sacadas. El artículo demuestra el potencial de los modelos basados en visión profunda para interpretar señales oculares complejas.

D. Artículo 4

Shubi, O., Meiri, Y., Hadar, C. A., And Berzak, Y. (2024). "Fine-Grained Prediction of Reading Comprehension from Eye Movements." Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Segúnn lo indicado en [4] este estudio explora una tarea más avanzada: predecir el nivel de comprensión lectora a partir de los patrones de movimiento ocular registrados durante la lectura. Los autores recopilaron un conjunto de datos en el cual los participantes respondían preguntas de comprensión tras leer textos mientras se registraban sus fijaciones, duración de sacadas, regresiones y cambios pupilares.

El enfoque metodológico incluyó modelos de machine learning tradicionales y modelos multimodales basados en redes neuronales que combinaban texto y señales oculares. La evaluación se llevó a cabo mediante métricas como exactitud, AUC y macro-F1, utilizando validación cruzada sujeto-independiente para medir capacidad de generalización. Los resultados muestran que las señales oculares contienen información discriminativa suficiente para predecir el rendimiento en comprensión lectora, aunque la tarea presenta un nivel de dificultad considerable. El estudio respalda la idea de que los movimientos oculares permiten inferir procesos cognitivos de alto nivel, lo cual se alinea con los objetivos del análisis de relevancia en lecturas.

IV. ENTRENAMIENTO Y EVALUACION DE MODELOS

A. Configuración experimental

Para la evaluación de los modelos de clasificación se utilizó un esquema de validación estratificada, con el fin de preservar la proporción original de clases (0, 1 y 2) en cada partición. El conjunto de datos fue dividido en:

Esta división estratificada garantiza que el modelo no se vea afectado por un posible desbalance entre las clases y permite una evaluación más robusta del desempeño.

Adicionalmente, la validación interna de cada modelo se realizó con base en las métricas calculadas tanto para el conjunto de entrenamiento como para el conjunto de prueba, incluyendo reportes de clasificación, matrices de confusión y curvas ROC multiclase (One-vs-Rest y One-vs-One).

No fue necesario aplicar técnicas de sobremuestreo o submuestreo, dado que la distribución de clases en el dataset es relativamente balanceada (aprox. 27%, 30% y 43%).

Para todos los modelos se empleó un pipeline unificado que encapsula:

- Estandarización de variables numéricas (StandardScaler)
- Codificación de variables categóricas (OneHotEncoder con handle_unknown='ignore')
- Remoción de columnas no informativas (lineNo)

Esto asegura que todos los modelos reciban datos preprocesados de manera consistente, eliminando sesgos derivados de escalamiento o representación de categorías.

1) Modelos evaluados e hiperparámetros: Evaluamos cinco tipos de modelos exigidos por el enunciado, además de variantes adicionales (LDA y QDA) como se evidencia en la **TABLA II**.

Nota: Los hiperparámetros finales utilizados por el pipeline fueron almacenados en un archivo JSON generado por el notebook.

2) Métricas de evaluación y justificación: Dado que se trata de un problema multiclase (3 clases) con ligera variación en la distribución, se emplearon métricas que consideran desempeño global:

TABLA II
MODELOS IMPLEMENTADOS Y SUS HIPERPARÁMETROS ANALIZADOS

Tipo de Modelo	Modelo Implementado	Hiperparámetros Analizados
Paramétrico	Regresión Logística Multiclase	`C`, `solver`, `class_weight`, `max_iter`, `multi_class`
Paramétrico (discriminantes)	LDA, QDA	`solver`, `shrinkage` (LDA), `reg_param` (QDA)
No Paramétrico	K-Nearest Neighbors	`n_neighbors`, `metric`, `p`, `weights`
Ensamble de Árboles	Random Forest	`n_estimators`, `max_depth`, `max_features`, `class_weight`
Red Neuronal Artificial	MLPClassifier	`hidden_layer_sizes`, `activation`, `solver`, `learning_rate`, `alpha`, `max_iter`, `batch_size`
Máquina de Vectores de Soporte	SVC (kernel RBF)	`C`, `gamma`, `kernel`, `class_weight`, `probability`

- **Accuracy:** Nos brinda la medida general del porcentaje de predicciones correctas.
- **Precision, Recall y F1-score (macro y weighted):**
  - **Macro:** cada clase pesa igual → útil si alguna clase es minoritaria.
  - **Weighted:** pondera según frecuencia → adecuado para medir desempeño global real.
- **Matrices de Confusión (Train y Test):** Nos permiten identificar confusiones entre clases.
- **Curvas ROC multiclase (OvR y OvO):** Utilizamos ROC_AUC_macro para evaluar separabilidad entre clases.

B. Resultados del entrenamiento del modelo

A continuación se resumen los resultados obtenidos durante la experimentación. Las métricas provienen directamente de las funciones implementadas (compute_metrics_multiclass) y del conjunto de prueba.

La siguiente tabla resume el rendimiento final de cada modelo en el conjunto de prueba:

Modelo	Accuracy	F1-score	Precision	Recall
Random Forest	**0.661**	**0.661**	**0.662**	**0.661**
KNN	0.598	0.599	0.602	0.598
SVC (RBF)	0.593	0.590	0.591	0.593
MLPClassifier	0.577	0.575	0.574	0.577
LDA	0.504	0.505	0.511	0.504
Logistic Regression	0.493	0.487	0.488	0.493
QDA	0.462	0.442	0.472	0.462

TABLA III
COMPARACIÓN DE MODELOS DE CLASIFICACIÓN UTILIZANDO MÉTRICAS GLOBALES.

1) Interpretación general de los resultados:

- **Modelos Paramétricos (LR, LDA, QDA)**
  - Presentaron el desempeño más bajo.
  - La Regresión Logística se vio limitada por la naturaleza no lineal del problema.
  - LDA funcionó mejor gracias al uso de shrinkage, pero aún limitado.
  - QDA mostró inestabilidad por matrices de covarianza no invertibles, evidenciado por warnings en el entrenamiento.

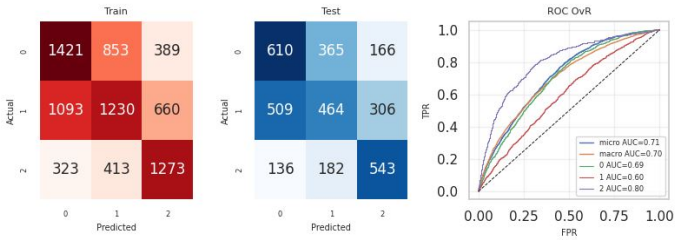

Fig. 2. Gráfica de modelo Regresión Logistica

Fig. 3. Gráfica de modelo Discriminante lineal

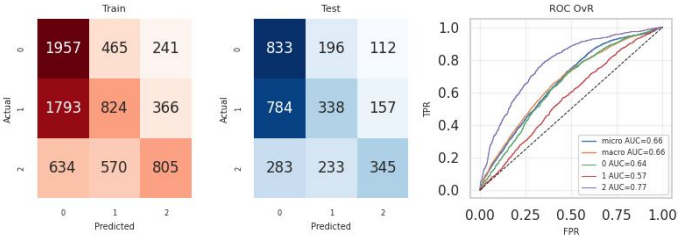

Fig. 4. Gráfica de modelo Discriminante cuadrático

- **Modelo No Paramétrico (KNN)**
  - Obtuvo resultados aceptables (60%)
  - Beneficiado por el escalamiento previo.
  - La distancia Manhattan (p=1) y k=3 permitieron captar relaciones locales en el espacio de características.

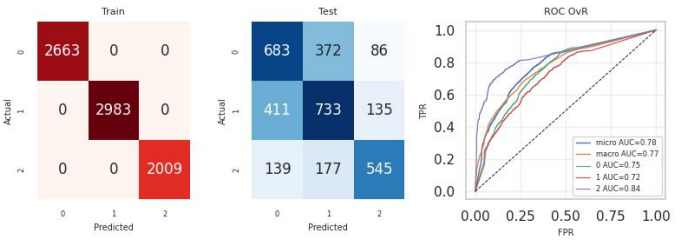

Fig. 5. Gráfica del modelo KNN

- **Modelo Ensamble (Random Forest)**
  - Mejor modelo del estudio.
  - Alto rendimiento y fuerte capacidad para manejar relaciones no lineales.

– Beneficiado por el manejo de interacciones y reducida varianza debido al ensamble.

Fig. 6. Gráfica del Modelo Random Forest

- **Red Neuronal MLP**
  - Mostró overfitting severo:
    - Train Accuracy 1.00
    - Test Accuracy 0.577
  - El modelo requiere:
    - Regularización más fuerte.
    - Menos capas.
    - Early stopping.

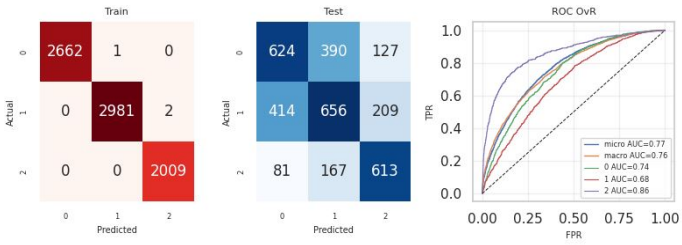

Fig. 7. Gráfica del modelo Red Neuronal MLP

- **Máquina de Vectores de Soporte (SVC)**
  - Logró desempeño competitivo (5̃9%).
  - Las curvas ROC mostraron buena separabilidad entre clases (AUC macro 0.78).
  - El kernel RBF capturó relaciones no lineales, pero con mayor costo computacional.

Fig. 8. Gráfica del modelo Máquina de vectores de soporte

Adjuntamos en el repositorio el archivo `hiperparametros_modelos.json`, el cual contiene la información de los hiperparámetros definidos para cada modelo.

2) Efecto de los Hiperparámetros: Durante las ejecuciones realizadas observamos lo siguiente:

- **Random Forest**

```
---- Tabla de Comparación Final ----
                                Accuracy  F1_Score  Precision    Recall  \
Regresion_Logistica_Multiclase  0.492838  0.486956   0.488327  0.492838
Regresion_Logistica_Multiclase_LDA  0.504115  0.505476   0.511103  0.504115
K-Nearest_Neighbors             0.597684  0.599054   0.602229  0.597684
Random_Forest_Classifier        0.661384  0.661431   0.662432  0.661384
MLPClassifier                   0.576958  0.575021   0.574251  0.576958
SVC                             0.593112  0.589528   0.590945  0.593112
Regresion_Logistica_Multiclase_QDA  0.462054  0.442075   0.471701  0.462054

                                                       Hiperparametros
Regresion_Logistica_Multiclase  {'C': 1.0, 'class_weight': 'balanced', 'dual':...
Regresion_Logistica_Multiclase_LDA  {'covariance_estimator': None, 'n_components':...
K-Nearest_Neighbors             {'algorithm': 'auto', 'leaf_size': 30, 'metric...
Random_Forest_Classifier        {'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
MLPClassifier                   {'activation': 'relu', 'alpha': 0.0001, 'batch...
SVC                             {'C': 5, 'break_ties': False, 'cache_size': 20...
Regresion_Logistica_Multiclase_QDA  {'priors': None, 'reg_param': 0.0, 'store_cova...
```

Fig. 9. Gráfica compración de los resultados

  - Aumentar **n_estimators** mejora la estabilidad.
  - Implementar **class_weight='balanced'** favoreció el desempeño en clases menos frecuentes.
- **SVC**
  - Hacer **metric='manhattan' y k=3** maximizaron el desempeño.
  - Valores mayores de k suavizaban demasiado las fronteras.
- **SVC**
  - Hacer **C=5** dio mejor margen entre bias/varianza.
  - Indicar **probability=True** era necesario para curvas ROC.
- **MLP**
  - Las configuraciones profundas inducen sobreajuste.
  - Se recomienda usar:
    - capas más pequeñas.
    - alpha mayor.
    - early_stopping=True.

3) Conclusiones de lo ejecutado: De la ejecución realizada obtenemos las siguientes conclusiones:

- Random Forest es el modelo más adecuado para este dataset.
- La naturaleza del problema exige modelos capaces de capturar interacciones no lineales, lo cual explica el bajo desempeño de modelos lineales.
- La inclusión de preprocesamiento (scaling + encoding) fue clave para SVM, KNN y MLP.
- Las métricas macro mostraron que ningún modelo favoreció de forma significativa una clase sobre otra.

V. REDUCCIÓN DE DIMENSIÓN

Finalmente procedemos a realizar el analisis y evaluación de diferentes tecnicas de reduccion de dimensión y así establecer que tan posible es realizar un reduccipon de la complejidad del modelo final.

A. Análisis individual de variables

El análisis individual de variables se emplea para evaluar la capacidad discriminativa de cada característica del conjunto de datos de movimientos oculares, permitiendo identificar atributos irrelevantes o con baja relación estadística con la variable objetivo. Este proceso es esencial para reducir ruido, evitar

sobreajuste y disminuir la complejidad del modelo, ya que características poco informativas pueden degradar el rendimiento predictivo y aumentar innecesariamente la dimensionalidad. Mediante métricas como *F-score* y *mutual information*, es posible determinar qué variables aportan valor al proceso de clasificación y cuáles son candidatas a ser eliminadas, estableciendo así una base sólida para etapas posteriores de selección y reducción de características como PCA o UMAP.

```
=== ANÁLISIS POR VARIABLE ORIGINAL (tras preprocesado) ===
Variables originales analizadas: 26
Umbral F_score_max (q=0.25): 6.4385
Umbral MI_max (q=0.25): 0.0042

VARIABLES CANDIDATAS A ELIMINAR (n=2):
                  F_score_max    MI_max  n_columnas_codificadas
variable_original
nextFixDur           0.201882  0.002198                       1
landingPos           2.065258  0.001362                       1
```

Fig. 10. Gráfica resultados de análisis de variables individuales

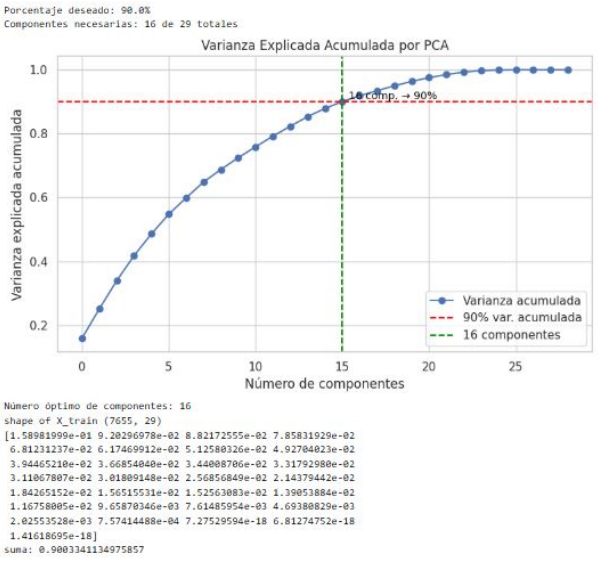

Fig. 11. Gráfica resultados de varianza acumulada

B. Extracción de características lineal - PCA

Es una técnica estadística de reducción de dimensionalidad que transforma un conjunto de variables numéricas en un espacio de menor dimensión mediante la identificación de nuevas características llamadas componentes principales, las cuales capturan la mayor cantidad posible de variabilidad del conjunto de datos. Este método permite detectar y descartar variables con poca aportación informativa, reduciendo ruido y preservando la estructura esencial de los datos mediante combinaciones lineales de las variables originales que resultan entre sí incorrelacionadas. *PCA*, al ser un método no supervisado que opera únicamente sobre las características y no sobre las etiquetas, requiere que los datos sean normalizados previamente y selecciona los componentes en función de su varianza explicada, ordenando de mayor a menor importancia cada dirección del espacio. En esencia, el procedimiento consiste en estandarizar los datos, calcular los autovalores y autovectores de la matriz de covarianza, seleccionar los k componentes más representativos y proyectar el conjunto original sobre este nuevo subespacio, logrando así una representación compacta y más eficiente para el entrenamiento de modelos de aprendizaje automático.

	Accuracy	F1_Score	Precision	Recall	\
Regresion_Logistica_Multiclase	0.492838	0.486956	0.488327	0.492838	
Regresion_Logistica_Multiclase_LDA	0.504115	0.505476	0.511103	0.504115	
K-Nearest_Neighbors	0.597684	0.599054	0.602229	0.597684	
Random_Forest_Classifier	0.661384	0.661431	0.662432	0.661384	
MLPClassifier	0.576958	0.575021	0.574251	0.576958	
SVC	0.593112	0.589528	0.590945	0.593112	
Regresion_Logistica_Multiclase_QDA	0.462054	0.442075	0.471701	0.462054	
Random_Forest_Classifier_PCA	0.552880	0.552716	0.558095	0.552880	
K-Nearest_Neighbors_PCA	0.536422	0.537148	0.540307	0.536422	

	Hiperparametros
Regresion_Logistica_Multiclase	{'C': 1.0, 'class_weight': 'balanced', 'dual':...
Regresion_Logistica_Multiclase_LDA	{'covariance_estimator': None, 'n_components':...
K-Nearest_Neighbors	{'algorithm': 'auto', 'leaf_size': 30, 'metric...
Random_Forest_Classifier	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
MLPClassifier	{'activation': 'relu', 'alpha': 0.0001, 'batch...
SVC	{'C': 5, 'break_ties': False, 'cache_size': 20...
Regresion_Logistica_Multiclase_QDA	{'priors': None, 'reg_param': 0.0, 'store_cova...
Random_Forest_Classifier_PCA	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
K-Nearest_Neighbors_PCA	{'algorithm': 'auto', 'leaf_size': 30, 'metric...

Fig. 12. Gráfica resultados de análisis Extracción de características lineal - PCA

C. Extracción de características no lineal - UMAP

Es una técnica avanzada de reducción de dimensionalidad basada en principios de teoría de grafos y geometría diferencial, cuyo objetivo es preservar tanto la estructura local como global de los datos al proyectarlos en un espacio de menor dimensión. A diferencia de técnicas lineales como PCA, UMAP captura relaciones no lineales y complejas entre las muestras, generando representaciones más compactas y útiles para clasificación, visualización o compresión de datos. UMAP construye un grafo de proximidad en el espacio original y lo optimiza para obtener un grafo equivalente en el espacio reducido, manteniendo la coherencia de las distancias entre puntos. Este enfoque permite conservar patrones intrínsecos de alta dimensionalidad, mejorar la separabilidad entre clases y reducir ruido sin perder información relevante. Asimismo, UMAP puede operar de manera supervisada incorporando las etiquetas para reforzar la estructura discriminativa de los datos, lo que mejora el desempeño posterior de los modelos de aprendizaje automático. Su flexibilidad y capacidad para modelar distribuciones complejas lo convierten en una herramienta poderosa en análisis exploratorio y preprocesamiento avanzado.

	Accuracy	F1_Score	Precision	Recall	\
Regresion_Logistica_Multiclase	0.492838	0.486956	0.488327	0.492838	
Regresion_Logistica_Multiclase_LDA	0.504115	0.505476	0.511103	0.504115	
Regresion_Logistica_Multiclase_QDA	0.462054	0.442075	0.471701	0.462054	
K-Nearest_Neighbors	0.597684	0.599054	0.602229	0.597684	
Random_Forest_Classifier	0.661384	0.661431	0.662432	0.661384	
MLPClassifier	0.576958	0.575021	0.574251	0.576958	
SVC	0.593112	0.589528	0.590945	0.593112	
Random_Forest_Classifier_PCA	0.552880	0.552716	0.558095	0.552880	
K-Nearest_Neighbors_PCA	0.536422	0.537148	0.540307	0.536422	
Random_Forest_Classifier_UMAP	0.661384	0.661431	0.662432	0.661384	
K-Nearest_Neighbors_UMAP	0.597684	0.599054	0.602229	0.597684	

	Hiperparametros
Regresion_Logistica_Multiclase	{'C': 1.0, 'class_weight': 'balanced', 'dual':...
Regresion_Logistica_Multiclase_LDA	{'covariance_estimator': None, 'n_components':...
Regresion_Logistica_Multiclase_QDA	{'priors': None, 'reg_param': 0.0, 'store_cova...
K-Nearest_Neighbors	{'algorithm': 'auto', 'leaf_size': 30, 'metric...
Random_Forest_Classifier	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
MLPClassifier	{'activation': 'relu', 'alpha': 0.0001, 'batch...
SVC	{'C': 5, 'break_ties': False, 'cache_size': 20...
Random_Forest_Classifier_PCA	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
K-Nearest_Neighbors_PCA	{'algorithm': 'auto', 'leaf_size': 30, 'metric...
Random_Forest_Classifier_UMAP	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...
K-Nearest_Neighbors_UMAP	{'algorithm': 'auto', 'leaf_size': 30, 'metric...

Fig. 13. Gráfica resultados de análisis Extracción de características no lineal - UMAP

D. Conclusiones

Los resultados obtenidos permiten concluir que los modelos no paramétricos, especialmente *Random Forest*, presentan el mejor desempeño en la clasificación de movimientos oculares dentro del conjunto de datos utilizado. Su rendimiento superior se refleja de manera consistente en las métricas de *Accuracy*, *F1-score*, *Precision* y *Recall*, superando tanto a los modelos paramétricos tradicionales (Regresión Logística, LtextscLDA, QDA) como a los modelos basados en redes neuronales y máquinas de soporte vectorial. Además, el análisis de reducción de dimensionalidad mediante *PCA* permitió identificar que el espacio original contenía variables con baja capacidad discriminativa, logrando conservar aproximadamente el 90% de la varianza con un número reducido de componentes. Sin embargo, la aplicación posterior de *PCA* a los mejores modelos no produjo mejoras significativas en el rendimiento, lo que sugiere que estos algoritmos ya manejaban adecuadamente la estructura de los datos en su espacio original.

Por otro lado, la reducción de dimensionalidad mediante *UMAP* demostró ser útil para explorar la estructura interna de los datos y mejorar la visualización, aunque no generó mejoras claras en el rendimiento de los modelos cuando se utilizó como etapa previa al entrenamiento supervisado. *UMAP* permitió capturar relaciones no lineales y patrones de agrupamiento, pero al integrarlo en el proceso de clasificación no se apreciaron incrementos sustanciales frente a los modelos entrenados sin reducción dimensional. En conjunto, el estudio confirma que la combinación del preprocesamiento adecuado, el análisis individual de variables y la evaluación sistemática de modelos permite identificar estrategias óptimas de clasificación, concluyendo que *Random Forest* sigue siendo la técnica más robusta y efectiva para este tipo de datos, incluso frente a métodos avanzados de reducción de dimensionalidad como *PCA* y *UMAP*.

Referencias

[1] J. Salojärvi, J. Puolamäki, and S. Kaski, “Inferring relevance from eye movements: Feature extraction,” in *Eye Tracking Challenge 2005*, Helsinki, Finland, 2005. [Online]. Available: https://research.ics.aalto.fi/events/eyechallenge2005/

[2] A. Papoutsaki, P. Sangkloy, J. Laskey, N. Daskalova, J. Huang, and J. Hays, “WebGazer: Scalable webcam eye tracking using user interactions,” in *Proc. 25th Int. Joint Conf. Artif. Intell. (IJCAI)*, New York, NY, USA, 2016. [Online]. Available: https://cs.brown.edu/people/apapouts/

[3] N. Bhattacharya, S. Rakshit, J. Gwizdka, and P. Kogut, “Relevance prediction from eye-movements using semi-interpretable convolutional neural networks,” *arXiv preprint arXiv:2001.05152*, 2020. [Online]. Available: https://arxiv.org/abs/2001.05152

[4] O. Shubi, Y. Meiri, C. A. Hadar, and Y. Berzak, “Fine-grained prediction of reading comprehension from eye movements,” in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024. [Online]. Available: https://aclanthology.org/2024.emnlp-main.198/