

# Proyectos Aprendizaje Computacional - 2025

Eduardo Morales, Hugo Jair Escalante

September 24, 2025

Esta es una lista de posibles proyectos para el curso de Aprendizaje Computacional. Los proyectos pueden hacerse en cualquier lenguaje y sistema operativo. Si tienen algún proyecto en mente, también lo podemos considerar. Se van a realizar dos presentaciones de los proyectos. Una de avances, en donde se espera ver cómo van con sus proyectos y poderles dar retro-alimentación, y una presentación final en donde ya se va a calificar el proyecto. También se espera que entreguen un reporte al final del curso sobre su proyecto en formato de artículo tipo LNCS en inglés.

## 1 Proyectos 2025

\* **Árboles de decisión considerando distancia entre clases:** Uno de los elementos clave en la inducción de árboles de decisión es el criterio de selección de atributos. Recientemente han surgido propuestas que toman en cuenta la distancia entre clases basada en la medida Rao-Stirling. El proyecto consiste en re-implementar con alguna mejora el algoritmo descrito en: S. Lee, C. Lee, K.G. Mun, D. Kim. Decision Tree Algorithm Considering Distances Between Classes. IEEE Access (<https://ieeexplore.ieee.org/document/9810212/>). En ese artículo calculan el centroide de cada clase y miden la distancia Euclideana entre centroides. Una posibilidad es ver si cada clase se puede agrupar en un sólo cluster y si no, ver cómo cambiar el algoritmo en caso de que existe más de un cluster por clase.

\* **Clasificación con reglas usando agregación Bayesiana:** El aprendizaje de reglas de clasificación es probablemente el modelo de aprendizaje más explicable de ML. Recientemente se propuso un algoritmo basado en ensambles y lo que llaman máquina puntual de Bayes (*Bayes point machine*). Ver: L. Bergamina, M. Polato, F. Aiolli. Improving rule-based classifiers by Bayes point aggregation. Neurocomputing 612 (2025). (<https://www.sciencedirect.com/science/article/pii/S0925>) La idea sería primero implementar un ensamble de clasificadores, alterando el orden de los ejemplos, y tratar de re-implementar lo descrito en el artículo con alguna mejora.

\* **Aprendizaje semi-supervisado de galaxias** Existen diferentes bases de datos de galaxias, sin embargo, sólo un porcentaje pequeño de ellas han sido clasificadas. En el proyecto se plantea usar aprendizaje semi-supervisado para construir un mejor clasificador de galaxias.

\* **Modelado de supervivencia para datos financieros:** El modelado de supervivencia (SM) consiste en predecir el tiempo transcurrido hasta que ocurre un evento específico. El SM tiene un inmenso valor en el sector financiero, donde el análisis implica el seguimiento de un grupo de entidades desde un evento índice hasta un evento de resultado designado. Por ejemplo, el evento índice podría ser la originación de un préstamo, y el evento de resultado su reembolso o impago. En este proyecto se plantea desarrollar una solución al *FinSurvival Challenge*, donde el objetivo principal es la predicción del tiempo transcurrido hasta un evento. El proyecto involucra:

- Familiarizarse con el problema, datos y plataforma de trabajo del [FinSurvival Challenge](#).
- Entendimiento y ejecución del baseline de la tarea.
- Desarrollo de solución propuesta.
- Estudios de ablación y evaluación, incluyendo comparación con el estado-del-arte.

\* **Ensamblados de *manifold learners* para análisis de datos:** Existen muchos métodos para el aprendizaje espacios embebidos (topologías, *manifolds*) que permiten reducir la dimensionalidad de datos tabulares. Este proyecto explora el efecto que tendrá la combinación

de diversos métodos de *manifold learning* para abordar tareas clásicas como visualización, agrupamiento y clasificación de datos. El proyecto plantea:

- Conseguir implementaciones de métodos diversos para *manifold learning*, incluyendo: PCA, LDA, LLE, ISOMAP, TSNE, etc. Un buen inicio es hojear algunos capítulos de [Elements of Dimensionality Reduction and Manifold Learning](#).
  - Seleccionar conjuntos de datos estándar para evaluación en las tareas de visualización, clasificación y agrupamiento.
  - Desarrollo de método de fusión de *manifolds*. Un baseline sería la aplicación anidada.
  - Evaluación y estudios de ablación en las tareas seleccionadas.
- \* **Encadenamiento de métodos de ingeniería de características:** La *feature engineering* (FE) comprende técnicas para generación y selección de atributos con el objetivo de mejorar el performance de tareas de aprendizaje supervisado en datos tabulares. Existen muchos métodos y librerías para abordar el problema. El proyecto plantea la aplicación secuencial de estos métodos para determinar si el uso encadenado es capaz de proveer atributos más discriminativos que usar un método de forma aislada. El proyecto comprende:
- Selección de conjuntos de datos estándar de alta dimensionalidad.
  - Selección de técnicas de FE que habrán de considerarse en el estudio. Un buen inicio es [How Usable is Automated Feature Engineering for Tabular Data?](#).
  - Desarrollo de la estrategia de encadenamiento de métodos de FE.
  - Evaluación, y estudios de ablación.
- \* **Búsqueda de respuestas en documentos digitalizados:** A pesar de los avances en digitalización de documentos, existen problemas abiertos que tienen gran aplicación en la industria. Uno de ellos es la búsqueda de respuestas (VQA) a partir de documentos no estructurados - digitalizados. Se plantean algunos proyectos en esta temática, que involucran lo siguiente:
- Construcción de recursos para evaluación.
  - Uso de métodos de OCR
  - Implementación de técnicas de procesamiento de lenguaje natural básica.
  - Uso de modelos de lenguaje y visión.

Para tener una idea de las expectativas del proyecto, se puede revisar: [Enhancing Document VQA Models via Retrieval-Augmented Generation](#). Para mayor información acercarse a los instructores.

- \* **LWL para el reconocimiento de emociones en perros:** El aprendizaje localmente ponderado (*Locally weighted learning, LWL*) tiene, entre otras ventajas, el poder lidiar con problemas de clasificación de grano fino. En este proyecto se plantea el desarrollo de métodos de métodos de LWL para el reconocimiento de emociones en imágenes de perros. Plan tentativo:
1. Obtener datos y modelo base para el reconocimiento de emociones en perros ([F. Hernandez et al. Dog emotion recognition from images in the wild: DEBIw dataset and first results. ACI, 2022.](#) ).
  2. Estudio, diseño y desarrollo del método de LWL que habrá de implementarse. Se sugiere un método basado en prototipos.
  3. Desarrollo del método, evaluación y comparación con el estado del arte.
- \* **Ordenamiento de muestras para el reconocimiento de emociones en perros:** El problema de reconocer emociones en perros a partir de imágenes es complejo por la diversidad de razas, variedad de fondos y escenarios. Una forma de mejorar el desempeño de modelos de clasificación es mediante el ordenamiento de muestras para entrenamiento. El proyecto plantea explorar el ordenamiento de muestras para reconocer emociones en perros.

1. Obtener datos y modelo base para el reconocimiento de emociones en perros ([F. Hernandez et al. Dog emotion recognition from images in the wild: DEBIw dataset and first results. ACI, 2022.](#) ).
2. Selección de métodos para ordenamiento de datos, se sugiere considerar las variantes de [Data distribution-based curriculum learning](#)
3. Desarrollo de método basado en ordenamiento, comparación con estado del arte

\* **Detección de memes inapropiados con aprendizaje basado en prototipos y prompting:**

Los memes pueden ser divertidos y reflejan eventos y situaciones culturales. Sin embargo, muchos de ellos contienen información ofensiva, que pretende denostar a ciertos individuos o grupos. Se propone abordar el problema de detectar discurso de odio en memes usando ejemplos prototípicos para instruir a grandes modelos de lenguaje.

1. Revisar la tarea número 3 y obtener los datos de la competencia [DIMEMEX 2025](#).
2. Desarrollar un método para selección de prototipos multimodales (identificar memes prototípicos).
3. Seleccionar un conjunto reducido de memes usando el método en 2) y usar esos ejemplos para instruir a un LLM para reconocer memes ofensivos. El prompting se puede basar en [.Improved Baselines with Visual Instruction Tuning](#)
4. Evaluación del método desarrollado y estudios de ablación variando el número de prototipos.

\* **Clasificación cross-dominio de Memes ofensivos:** Los memes pueden ser divertidos y reflejan eventos y situaciones culturales. Sin embargo, muchos de ellos contienen información ofensiva, que pretende denostar a ciertos individuos o grupos. Este proyecto plantea

1. Revisar literatura y obtener múltiples bases de datos para el estudio de memes ofensivos. [La tesis de Horacio Vasquez](#) es un buen inicio.
2. Diseñar un método para aprendizaje cross-dominio (mezclando datos de diferentes bases de datos)
3. Evaluación del método: comparación con estado del arte y estudios de ablación

\* **Agentes basados en LLMs para etiquetado de memes ofensivos:** Los memes pueden ser divertidos y reflejan eventos y situaciones culturales. Sin embargo, muchos de ellos contienen información ofensiva, que pretende denostar a ciertos individuos o grupos. El proyecto plantea lo siguiente:

- Entender y reproducir el enfoque de [MIND:AMulti-agent Framework for Zero-shot Harmful Meme Detection](#). De particular interés es la estimación de distancia con base de datos (Eq. (2)).
- Implementar una unidad tipo GMU para calcular la distancia con ejemplos relevantes (en lugar de plain coseno).
- Evaluación y comparación con el estado del arte.

## 2 Proyectos pasados

1) **Ensamblados de árboles de decisión:** Los ensambles de clasificadores generalmente funcionan con algoritmos inestables, como los árboles de decisión, los cuales cambian sus modelos con pequeños cambios en los datos. Por otro lado, generalmente se aprenden varios árboles (e.g., 100) aunque no sean necesarios todos. La idea es crear un ensamble en donde se detenga la generación de nuevos árboles cuando dejen de ser útiles para el desempeño del ensamble.

1. Entender cómo funciona Random Forest
2. Diseñar un esquema de parada de generación de árboles basado en desempeño y estancamiento
3. Hacer pruebas en varias bases de datos (e.g., UCI) comparando Random Forest y Random Forest con parada temprana en términos de desempeño y tamaño de los árboles.

**2) Aprendizaje semi supervisado:** Los algoritmos semi-supervisados en ocasiones no reportan los resultados esperados. Esto puede ser porque los ejemplos etiquetados y no etiquetados no provienen de la misma distribución o porque los modelos usados no son adecuados para representar los modelos. La idea es crear un algoritmo de aprendizaje semi-supervisado que detecte de forma temprana cuándo puede fallar y corregir o desechar a los ejemplos no etiquetados.

1. Implementar un algoritmo de aprendizaje semi-supervisado (lo más directo es usar self-training)
2. Crear un mecanismo que detecte de forma temprana cuándo el algoritmo no está funcionando
3. Cambiar el clasificador base y volver a probar

Una posibilidad es usar SVM como clasificador base e ir cambiando los kernels de acuerdo al desempeño del sistema.

**3) Datos desbalanceados:** Los clasificadores muchas veces se ven afectados cuando existe un desbalance en las clases. Lo más común es realizar un sobre/sub-muestreo para balancear las clases, sin embargo, el sobre o sub-muestreo sólo se debe de realizar mientras no se tengan resultados aceptables, La idea es hacer un algoritmo de clasificación con datos desbalanceados, en donde se haga sobremuestreo o un submuestreo sólo si es necesario, y hacerlo sólo para que se tengan resultados aceptables (i.e., no se tienen porque balancear las clases)

1. Implementar un algoritmo de balanceo de clases (puede ser ROS, RUS, Smote, etc.)
2. Idear un mecanismo que detecte la cantidad de muestreo a realizar con base en unos cuantos experimentos
3. Conjuntar este mecanismo en un algoritmo y probarlo contra otros algoritmos que realizan o no muestreo, en términos de desempeño y cantidad de datos generados.

**4) Co-clasificación usando "multi-view":** Los algoritmos de *Multi-view* o múltiples vistas, toman varias representaciones del mismo problema para realizar la clasificación. La idea es construir clasificadores más robustos cuando se tiene información complementaria.

1. Buscar bases de datos de *multi-view learning* y algoritmos
2. Hacer embedding de cada representación y juntarlas en un solo clasificador
3. Crear un clasificador para cada representación y juntar sus resultados en un meta-clasificador
4. Comparar ambos enfoques

**5) Feature construction:** Los atributos usados en los algoritmos de clasificación no siempre son los más adecuados. Una posibilidad es crear nuevos atributos usando los atributos existentes.

1. Diseñar un esquema de selección de los mejores atributos y mecanismos para combinarlos
2. Crear un algoritmo que construya automáticamente nuevos atributos usando combinaciones de los existentes
3. Probar el algoritmo en diferentes bases de datos (e.g., UCI).

**6) Aprendizaje de redes bayesianas:** Los algoritmos de aprendizaje de redes bayesianas generalmente usan una estrategia *greedy* la cual puede caer en un mínimo local. Una posibilidad para hacer más exploración es usar una estrategia Monte Carlo en donde se prueban muchas posibles estructuras.

1. Diseñar un algoritmo de aprendizaje de redes bayesianas usando ideas de Monte Carlo Tree Search (MCTS)
2. Probar el algoritmo en varias bases de datos y compararlo contra estrategias tradicionales

7) **Reglas de asociación con acción:** Las reglas de asociación se han empleado para encontrar patrones interesantes y para clasificar, con buenos resultados. Sin embargo, no se han aplicado en ambientes dinámicos que involucren la ejecución de ciertas acciones.

1. Crear un algoritmo que aprenda reglas del tipo: IF <item-set> AND <acción> THEN <item-set> En donde las condiciones y consecuencias se construyan siguiendo esquemas parecidos a lo que hace a priori.
2. Crear un conjunto de secuencias de acciones para resolver un problema secuencial sencillo
3. Aprender reglas de ese conjunto y probar qué tan bien puede resolver el problema secuencial

8) **Incremental Learning:** La mayoría de los clasificadores aprenden un modelo para distinguir entre un conjunto fijo de clases. Recientemente se han estado desarrollando algoritmos para ir incrementando el número de clases. El proyecto plantea implementar un algoritmo de aprendizaje incremental para clasificar objetos.

1. Implementar un sistema de aprendizaje incremental (evitando lo que se conoce como *catastrophic forgetting*) buscando alguna mejora
2. Probarlo con imágenes de objetos de un ambiente tipo casa u oficina y que puedan servirle a un robot móvil
3. Probarlo con varios objetos extraídos de imágenes

Una posibilidad es usar un cluster jerárquico para organizar las clases y crear clasificadores binarios en los nodos intermedios.

9) **Prompting de grandes modelos de lenguaje para la fusión de información:** Los grandes modelos de lenguaje (LLMs) han causado revuelo y se están usando cada vez más para resolver problemas complejos (e.g., optimización y aprendizaje modelos). Este proyecto plantea el desarrollo de instrucciones ad-hoc (*prompting*) en LLMs para aprender a combinar listas de resultados (e.g., predicciones de modelos o resultados de sistemas de recuperación). Posibles actividades:

1. Entender cómo funciona un LLM, y estudiar técnicas de *prompting* para resolver problemas complejos (e.g., optimización, automl)
2. Obtener listas de resultados de diferentes dominios para fusionar información.
3. Desarrollo de método de prompting y evaluación de resultados.
4. Comparación con métodos de fusión de información convencionales.

10) **Aprendizaje de curricula (*curriculum learning*) para clasificación de textos:** El aprendizaje basado en curricula (CL) permite manipular el orden en que se administran los ejemplos a un método de aprendizaje con el objetivo de generar modelos con mejor desempeño, o bien, avanzar una convergencia más rápida que con aprendizaje convencional. Se plantea en este proyecto el desarrollo de un método de CL para la clasificación de textos en el contexto de detección de menciones de eventos violentos en Twitter. Plan tentativo:

1. Obtener datos y modelo base de la tarea de detección de eventos violentos (DAVINCIS) disponibles en: <https://codalab.lisn.upsaclay.fr/competitions/11312>
2. Definición de curricula para aprendizaje en problema a abordar.
3. Desarrollo del método de CL para abordar la tarea.
4. Evaluación y comparación con métodos del estado del arte.

11) **LWL para el reconocimiento de emociones en perros:** El aprendizaje localmente ponderado (*Locally weighted learning, LWL*) tiene, entre otras ventajas, el poder lidiar con problemas de clasificación de grano fino. En este proyecto se plantea el desarrollo de métodos de LWL para el reconocimiento de emociones en imágenes de perros. Plan tentativo:

1. Obtener datos y modelo base para el reconocimiento de emociones en perros <https://dl.acm.org/doi/abs/10.1145/3565995.3566041>

2. Estudio, diseño y desarrollo del método de LWL que habrá de implementarse
3. Desarrollo del método, evaluación y comparación con el estado del arte

**12) Agrupamiento de múltiples vistas para video:** En algunas aplicaciones, los datos pueden ser descritos por múltiples modalidades de información, por ejemplo un video puede describirse por información visual, de audio e incluso texto (ASR). En este proyecto se plantea el análisis no supervisado de un conjunto de datos de videos mediante agrupamiento de múltiples vistas. Plan tentativo:

1. Obtener la base de videos de comic mischief detection <https://github.com/hugojair/mocca-at-icmi21>
2. Estudio, y selección del método de agrupamiento de múltiples vistas (hint: revisar un survey reciente)
3. Desarrollo del método, evaluación y comparación con el estado del arte
4. Visualización de resultados usando PCA o T-SNE

**13) Programación genética para el aprendizaje de kernels:** Los métodos de *multiple kernel learning* se encargan de combinar kernels independientes para mejorar el desempeño de métodos basados en kernels (e.g., SVM). Este proyecto plantea el uso de programación genética (algoritmo evolutivo usando árboles como representación) para aprender una función que combine kernels y puedan usarse en un método de MKL. Plan tentativo:

1. Entender qué es MKL y familiarizarse con programación genética (GP)
2. Seleccionar datos de algún repositorio de ML (UCI, OpenML, kaggle, etc.)
3. Diseño y desarrollo del método basado en GP para el aprendizaje de funciones de fusión
4. Implementación y evaluación del método desarrollado