# Preguntas Abiertas y de Investigación

Examen intermedio de Reconocimiento de Patrones, parte 2

2025-2

## Pregunta 1: Regularización Adaptativa Híbrida

**Contexto:** En regresión lineal, la regularización $L_1$ (*Lasso*) ayuda a reducir las características al forzar a los coeficientes muy pequeños a 0, pero puede ser inestable cuando hay características correlacionadas. La regularización $L_2$ (*Ridge*) es estable pero no ayuda a reducir características. *Elastic Net* combina ambas linealmente: $\lambda_1\|w\|_1 + \lambda_2\|w\|_2^2$.

**a) Análisis Teórico (50%):** Diseña una nueva forma de regularización llamada *Adaptive Correlation-Aware Regularization* que ajuste dinámicamente entre $L_1$ y $L_2$ basándose en la estructura de correlación de las características. Específicamente:

- Propón una función de penalización $R(w, X)$ que tome en cuenta la matriz de correlación de características
- Para características altamente correlacionadas entre sí, el término debe comportarse más como $L_2$ (para estabilidad)
- Para características independientes, debe comportarse más como $L_1$ (para reducción de dimensionalidad)
- Deriva matemáticamente el gradiente de tu función de regularización
- Discute bajo qué condiciones tu regularización sería equivalente a $L_1$ puro o $L_2$ puro

**b) Implementación y Validación Empírica (50%):** Implementa tu algoritmo usando descenso de gradiente y compáralo contra *Lasso*, *Ridge* y *Elastic Net* en los siguientes escenarios sintéticos:

1. Conjunto de datos con 3 grupos de características: cada grupo tiene 10 características altamente correlacionadas ( $> 0.8$), pero los grupos son independientes entre sí. Solo 1 característica por grupo es realmente predictiva.
2. Conjunto de datos con 50 características completamente independientes, de las cuales solo 10 son predictivas.
3. Conjunto de datos con estructura de correlación gradual: características ordenadas donde características $i$ correlaciona con características $j$ como $\rho = \exp(-|i - j|/5)$.

Genera 1000 observaciones para cada escenario con ruido gaussiano. Compara:

1. MSE (entrenamiento y prueba)
2. Número de características seleccionadas (coeficientes no-cero)

**Entregables:**

1. Derivación matemática (PDF con LaTeX o markdown bien formateado)
2. Código documentado (puede estar en una libreta)
3. Libreta con experimentos, visualizaciones y comentarios

## Pregunta 2: Clustering con Restricciones Prácticas

**Contexto:** En muchas aplicaciones reales de clustering, existen restricciones que los algoritmos estándar no pueden manejar:

- **Must-link:** ciertos pares de puntos DEBEN estar en el mismo cluster
- **Cannot-link:** ciertos pares NO PUEDEN estar en el mismo cluster
- **Balance:** los clusters deben tener tamaños similares ($\pm 20\%$)
- **Capacidad:** ningún cluster puede tener más de K elementos

### a) Diseño Algorítmico (50%):

Desarrolla una variante de K-Means llamada *Constrained-Capacity K-Means* que respete simultáneamente:

- Restricciones de must-link y cannot-link
- Límites de capacidad por cluster

Describe tu algoritmo en pseudocódigo. Considera:

- ¿Cómo modificas el paso de asignación cuando un cluster alcanza su capacidad?
- ¿Cómo garantizas que no violas las restricciones de must/cannot-link?
- ¿Qué haces cuando hay conflictos irresolubles? (e.g., must-link entre 3 puntos pero solo 2 caben en el cluster más cercano)

### b) Implementación (50%):

Implementa tu algoritmo y demuestra que funciona en este escenario:

- 1000 puntos en 2D organizados en 5 clusters naturales con diferentes densidades
- 20 restricciones must-link aleatorias
- 30 restricciones cannot-link aleatorias (algunas pueden ser contradictorias con must-link)
- Capacidad máxima: 250 puntos por cluster
- Capacidad mínima: 150 puntos por cluster

Compara contra:

- K-Means estándar (ignora restricciones)
- Algún baseline razonable que se te ocurra

**Entregables:**

1. Documento con descripción del algoritmo (PDF con LaTeX o markdown bien formateado)
2. Código implementado con pruebas
3. Libreta con experimentos

## Pregunta 3: Diagnóstico Avanzado de Data Leakage

**Contexto:** Data leakage es uno de los problemas más sutiles en ML. Algunas formas son obvias, otras extremadamente difíciles de detectar.

**Tu tarea:**

**a) Taxonomía de Leakage (30%):**

Investiga y crea una taxonomía completa de tipos de data leakage:

- Leakage temporal
- Leakage por preprocesamiento
- Leakage por target encoding
- Leakage por features que contienen información futura
- Leakage por duplicación de registros
- Leakage por estratificación incorrecta
- (Y otros que descubras)

Para cada tipo, proporciona:

- Definición formal
- Ejemplo concreto
- Por qué es problemático
- Cómo detectarlo
- Cómo prevenirlo

**b) Metodología para evitar el Data Leakeage (40%):**

Desarrolla un checklist que, al analizar un pipeline de ML, ayude a detectar posibles fuentes de leakage.

Tu checklist debe:

1. Verificar el orden de operaciones en el pipeline
2. Verificar si hay operaciones de preprocesamiento antes del train/test split
3. Detectar si hay características con correlación sospechosamente alta con el target ($>0.95$)
4. Identificar si hay características con información temporal que podrían contener "futuro"
5. Detectar duplicación o casi duplicación de registros entre conjuntos de entrenamiento y validación
6. Verificar si operaciones de *encoding* usan información del conjunto de validación

**c) Casos de Estudio (30%):**

Crea una libreta con un *pipeline* que contengan diferentes tipos de leakage (sutiles, no obvios). Demuestra que:

1. El modelo tiene rendimiento artificialmente inflado
2. Tu checklist ayuda a detectar el (los) problema(s)
3. Al corregir el leakage, el rendimiento baja a un nivel realista

**Entregables:**

1. Documento con taxonomía completa (incluir referencias)
2. Checklist (PDF con LaTeX o markdown bien formateado)
3. Libreta con caso de estudio bien documentado

## Pregunta 4: Límites Teóricos del Aprendizaje

**Contexto:** La teoría del aprendizaje estadístico proporciona cotas superiores sobre el error de generalización, pero estas cotas suelen ser muy conservadores y poco informativos en la práctica.

**a) Análisis de la cota superior VC Clásico (20%):**

Considera la cota superior clásica derivada de la dimensión VC:

Con probabilidad al menos $1 - \delta$:

$$E_{out}(h^*) \leq E_{in}(h^*) + \sqrt{\frac{d(\log(2m/d) + 1) - \log(\delta/d)}{m}}$$

Donde:

- $E_{out}(h^*)$ es el error fuera de muestra
- $E_{in}(h^*)$ es el error en muestra (en el conjunto de entrenamiento)
- $d$ es la dimensión VC
- $m$ es el tamaño del dataset

Para tres modelos:

1. Clasificador lineal en $(R)^10$ (d=11)
2. Clasificador polinomial de grado 3 en $(R)^5$
3. Red neuronal con 2 capas ocultas de 50 neuronas cada una

Calcula:

- La dimensión VC teórica
- La cota superior para $m \in \{100, 1000, 10000\}$

**b) *Tight Bounds* para Casos Específicos (30%):**

Las cotas superiores de VC son universales pero flojas. Investiga y deriva cotas superiores más ajustados para casos específicos:

1. Para regresión lineal con regularización $L_2$, la cota superior puede expresarse en términos de la norma de los pesos. Deriva esta cota y explícala con tus palabras.
2. Para clasificadores lineales con margen $M$, existe una cota superior basada en el margen que es más informativa que la cota VC clásica. Deriva o investiga esta cota de margen.

**c) Experimento de Validación Empírica (50%):** Diseña un experimento exhaustivo donde:

1. Generas múltiples conjuntos de datos sintéticos con diferentes:
   - Tamaños ($m \in \{50, 100, 500, 1000, 5000\}$)
   - Dimensionalidades ($d \in \{2, 5, 10, 20, 50\}$)
   - Niveles de ruido ($\sigma \in \{0.1, 0.5, 1.0, 2.0\}$)
2. Para cada combinación, entrena modelos de diferente complejidad y mide:
   - Error de entrenamiento
   - Error de test (promediado sobre 100 train/test splits)
   - Las cotas superiores teóricas
3. Visualiza:
   - ¿Qué tan ajustados son las cotas? (relación cota entre error real)
   - ¿En qué régimen $(n, d)$ las cotas son más informativos?
   - ¿Cómo afecta el ruido a la validez de las cotas?
4. Propón una heurística práctica basada en tus hallazgos para estimar cuándo un modelo está en riesgo de overfitting sin necesidad de usar bounds teóricos formales.

**Entregables:**

1. Derivaciones matemáticas (PDF con LaTeX o markdown bien formateado)
2. Libreta con código de simulación, visualizaciones comprensivas (gráficos 3D de $m \times d \times error$) evaluaciones que respondan las preguntas y heurística propuesta.