Machine Learning Supervisado: Algoritmos de Clasificación

HE2: Consultoría Económica con IA Responsable

Santiago Neira & Catalina Bernal

Universidad de los Andes
Departamento de Economía

Febrero 2026

Agenda de hoy

1. Del Problema de Regresión al Problema de Clasificación
2. El Puente: Regresión Logística y Probit
3. K-Nearest Neighbors (KNN)
4. Support Vector Machines (SVM)
5. Comparación de Algoritmos

Agenda de hoy

Recordatorio: Aprendizaje Supervisado

El marco general:

$$Y = f(X) + \varepsilon \tag{1}$$

Objetivo: Encontrar $\hat{f}$ tal que $\hat{Y} = \hat{f}(X)$ sea una buena aproximación.

Dos grandes familias según la naturaleza de Y:

Regresión

- $Y \in \mathbb{R}$ (continua)
- Predecir ingreso, precio, temperatura
- Métricas: MSE, RMSE, R^2

Clasificación

- $Y \in \{C_1, C_2, \ldots, C_K\}$ (categórica)
- Predecir fraude, default, diagnóstico
- Métricas: Accuracy, Precisión, Recall

¿Por Qué Clasificación? Casos Reales en Economía

Clasificación está en todas partes:

- **Banca:** ¿Aprobar o rechazar una solicitud de crédito?
  - Input: Ingreso, historial, score, deuda
  - Output: Aprobar / Rechazar
- **Marketing:** ¿Este cliente comprará el producto?
  - Input: Edad, compras pasadas, navegación web
  - Output: Comprará / No comprará
- **Recursos Humanos:** ¿Este candidato durará más de 1 año?
  - Input: Educación, experiencia, test psicológicos
  - Output: Permanecerá / Renunciará
- **Política pública:** ¿Este programa social ayudará a esta familia?
  - Input: Características socioeconómicas
  - Output: Beneficiará / No beneficiará

El Problema de Clasificación

Clasificación Binaria: $Y \in \{0, 1\}$ o $\{-1, +1\}$

Ejemplos:

- ¿El cliente hará default en el crédito? (Sí/No)
- ¿El correo es spam? (Spam/No spam)
- ¿El paciente tiene la enfermedad? (Positivo/Negativo)
- ¿La transacción es fraudulenta? (Fraude/Legítima)

Clasificación Multiclase: $Y \in \{1, 2, \ldots, K\}$

Ejemplos:

- Reconocimiento de dígitos escritos (0-9)
- Clasificación de sentimiento (Positivo/Neutral/Negativo)
- Diagnóstico médico (Sano/Enfermedad A/Enfermedad B)

¿Qué Predice un Clasificador?

Dos tipos de salidas:

1. **Clase predicha (hard classification):**

$$\hat{Y} = \hat{f}(X) \in \{0, 1\} \tag{2}$$

   Ejemplo: "Esta transacción ES fraude"

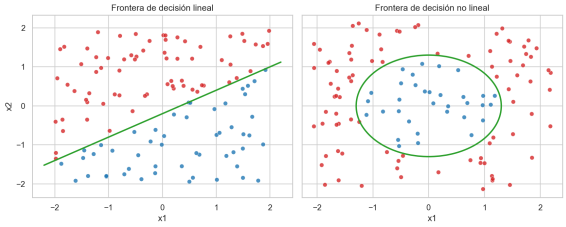
2. **Probabilidad de clase (soft classification):**

$$\hat{p}(Y = 1|X) = P(Y = 1|X = x) \tag{3}$$

   Ejemplo: "Esta transacción tiene 85 % de probabilidad de ser fraude"

   Luego aplicamos un umbral: $\hat{Y} = 1$ si $\hat{p} > 0{,}5$, sino $\hat{Y} = 0$

La Frontera de Decisión



Frontera de decisión: Región del espacio de features que separa las clases.

$$\text{Frontera} = \{x : P(Y = 1|X = x) = 0{,}5\} \tag{4}$$

- Modelos lineales: frontera es un hiperplano
- Modelos no lineales: frontera puede ser curva compleja

Agenda de hoy

Ya Conocen Este Modelo: Logit

En econometría: Modelo Logit (Logistic Regression)

$$P(Y_i = 1|X_i) = \frac{1}{1 + e^{-X_i'\beta}} = \frac{e^{X_i'\beta}}{1 + e^{X_i'\beta}} \tag{5}$$

O equivalentemente, en términos de log-odds:

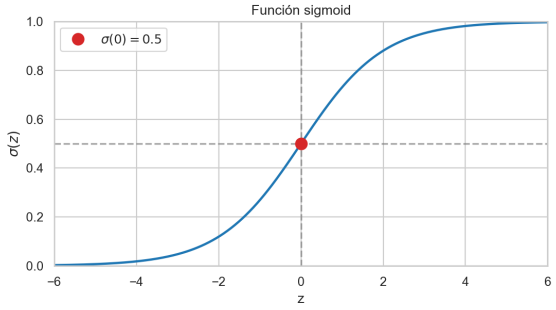
$$\log\left(\frac{P(Y = 1|X)}{1 - P(Y = 1|X)}\right) = X'\beta \tag{6}$$

Función logística (sigmoid):

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{7}$$

Mapea $\mathbb{R} \to (0, 1)$

La Función Sigmoid



Propiedades clave:

- $\lim_{z\to-\infty} \sigma(z) = 0, \quad \lim_{z\to\infty} \sigma(z) = 1$
- $\sigma(0) = 0{,}5$ (punto medio)
- Simétrica: $\sigma(-z) = 1 - \sigma(z)$
- Derivada: $\sigma'(z) = \sigma(z)(1 - \sigma(z))$

Estimación por Máxima Verosimilitud

No usamos OLS para logit. ¿Por qué?

- $Y \in \{0, 1\}$, no es continua
- Queremos estimar probabilidades $P(Y = 1|X) \in (0, 1)$

Máxima Verosimilitud:
Cada observación contribuye:

$$P(Y_i|X_i) = p_i^{Y_i}(1 - p_i)^{1-Y_i}, \quad \text{donde } p_i = \sigma(X_i'\beta) \tag{8}$$

Log-verosimilitud:

$$\ell(\beta) = \sum_{i=1}^{n} [Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i)] \tag{9}$$

Estimador: $\hat{\beta}_{MLE} = \arg\max_{\beta} \ell(\beta)$ (sin solución cerrada, requiere optimización numérica)

Dos Perspectivas, Un Modelo

Perspectiva Econométrica

- Interés en $\hat{\beta}_j$
- ¿Cuál es el efecto marginal de X_j?
- Interpretación de odds ratios
- Tests de hipótesis
- Inferencia causal

Perspectiva ML

- Interés en $\hat{p}(Y = 1|X)$
- ¿Qué tan bien clasificamos?
- Calibración de probabilidades
- Accuracy, precisión, recall
- Generalización

El cambio de paradigma

Econometría: "¿Aumentar X en 1 unidad cambia la probabilidad?"
ML: "¿Este nuevo cliente hará default?"

Interpretabilidad: Odds Ratios

Recordemos el modelo en términos de log-odds:

$$\log\left(\frac{P(Y=1|X)}{1-P(Y=1|X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \tag{10}$$

¿Qué son los "odds"?
Si $P(\text{default}) = 0{,}2$ (20 %), entonces:

$$\text{odds} = \frac{0{,}2}{0{,}8} = 0{,}25 = \text{"1 de cada 4"} \tag{11}$$

Odds Ratio (OR):

$$OR_j = e^{\beta_j} \tag{12}$$

- $OR = 2$: Duplica las odds (si eran $1/4$, ahora son $1/2$)
- $OR = 0{,}5$: Reduce las odds a la mitad

Ejemplo Económico: Default de Crédito

Contexto: Banco estima riesgo de default en tarjetas
Modelo estimado:

$$\log(\text{odds de default}) = -3 + 0{,}5 \cdot \text{deuda/ingreso} - 0{,}8 \cdot \text{score} + 1{,}2 \cdot \text{alta utilización} \tag{13}$$

Interpretación para comité de crédito:

- **Ratio deuda/ingreso** *Tipo: cociente (%) OR* $= 1{,}65$
  Un aumento de 1 unidad en el ratio (p. ej. 40 % → 50 %) incrementa los odds de default en 65 %. Refleja presión financiera estructural.
- **Score crediticio** *Tipo: variable continua OR* $= 0{,}45$
  Un aumento en el score reduce los odds de default en 55 %. Captura calidad histórica de pago.
- **Alta utilización de crédito** *Tipo: variable indicadora (0/1) OR* $= 3{,}32$
  Clientes con alta utilización tienen más de 3 veces los odds de default — señal de tensión financiera inmediata.

Lectura ejecutiva: Mayor presión financiera eleva riesgo; mejor historial crediticio lo mitiga.

Efectos Marginales: Más Intuitivo para Economistas

Problema con odds ratios: No son directamente probabilidades

Pregunta del tomador de decisiones:
"Si este cliente aumenta su ingreso en $1 millón, ¿cuánto baja su probabilidad de default?"

Efecto marginal:

$$\frac{\partial P(Y=1|X)}{\partial X_j} = \beta_j \cdot P(Y=1|X) \cdot [1 - P(Y=1|X)] \tag{14}$$

Depende del nivel base de probabilidad!

- Si $P = 0{,}5$: el efecto marginal es máximo
- Si $P = 0{,}01$ o $P = 0{,}99$: el efecto marginal es pequeño (curva S está aplanada)

En la práctica: Reportamos efectos marginales en el promedio (AME) o en valores específicos.

Logit en Política Pública: Un Ejemplo

Caso: ¿Qué determina la participación laboral femenina?

Variables:

- Educación (años)
- Edad
- Número de hijos
- Ingreso del cónyuge
- Acceso a jardín infantil

Pregunta de política:
"Si expandimos la cobertura de jardines infantiles en 10 %, ¿cuántas mujeres adicionales entrarían al mercado laboral?"

Necesitamos el efecto marginal de la variable jardín, no solo el odds ratio.

Interpretabilidad = Valor para la toma de decisiones

Los odds ratios son útiles para entender **dirección** y **magnitud relativa**. Los efectos marginales son útiles para **cuantificar impactos** en términos de probabilidades.

Regularización en Logit

Igual que en regresión lineal: podemos regularizar

Ridge Logistic Regression (L2):

$$\hat{\beta}^{Ridge} = \arg\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{15}$$

Lasso Logistic Regression (L1):

$$\hat{\beta}^{Lasso} = \arg\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{16}$$

- Previene overfitting cuando p es grande
- Lasso hace selección de variables
- Tuning de λ con cross-validation

Agenda de hoy

K-Nearest Neighbors: "Dime con Quién Andas..."

...y te diré quién eres

La idea más intuitiva en ML:

"Eres parecido a tus vecinos"

Ejemplos cotidianos:

- Si tus 5 amigos más cercanos son runners, probablemente tú también corres
- Si los 10 negocios cerca de tu local venden comida, tu local probablemente también
- Si las 3 empresas más parecidas a la tuya son tech startups, tú probablemente también

KNN formaliza esta intuición: Clasifica un nuevo punto según la clase mayoritaria de sus vecinos más cercanos.

K-Nearest Neighbors: La Idea

Algoritmo no paramétrico más simple:

Para clasificar un nuevo punto x_0:

1. Encontrar los K puntos de entrenamiento más cercanos a x_0
2. Llamar a este conjunto $\mathcal{N}_K(x_0)$ (sus "vecinos")
3. Asignar la clase por **voto mayoritario**:

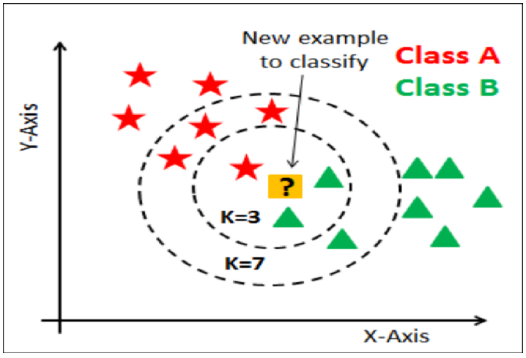
$$\hat{Y}(x_0) = \text{moda}\{Y_i : i \in \mathcal{N}_K(x_0)\} \tag{17}$$

O en términos de probabilidad:

$$\hat{P}(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_K(x_0)} \mathbb{I}(Y_i = j) \tag{18}$$

En palabras: La probabilidad de la clase es simplemente la proporción de vecinos que pertenecen a esa clase.

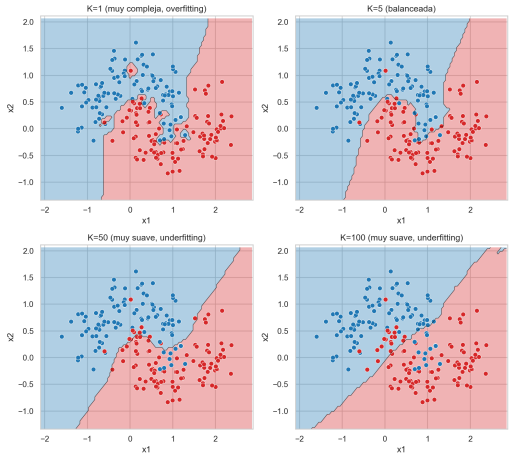
KNN: Ejemplo Visual



Interpretación:

- Con $K = 3$: Miramos solo los 3 vecinos más cercanos
- Con $K = 7$: Promediamos sobre más vecinos
- K pequeño: frontera compleja, más flexible
- K grande: frontera suave, menos flexible

KNN: Hiperparámetro K



Estática comparativa en KNN

K pequeño

- Alta varianza, bajo bias
- Overfitting

K óptimo

- Balance bias-varianza
- Generaliza bien

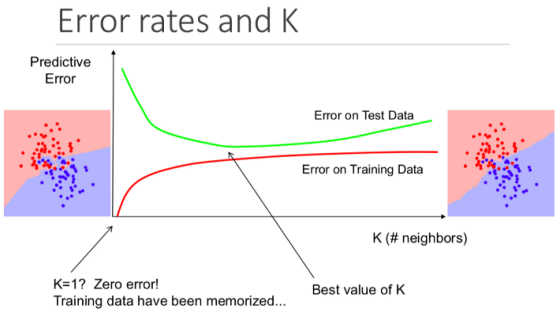
K grande

- Baja varianza, alto bias
- Underfitting

Selección de K: La Curva en U

Observaciones:

- Error de training siempre **aumenta** con K (menos flexible)
- Error de test tiene forma de **U**
- Elegimos K por cross-validation



KNN: Ventajas y Desventajas

Ventajas:

- Muy simple de entender e implementar
- No requiere entrenamiento (lazy learning)
- Naturalmente maneja fronteras no lineales
- Funciona para clasificación y regresión

Desventajas:

- Muy lento en predicción cuando n es grande
- Maldición de la dimensionalidad (alto p)
- Sensible a escalas de variables
- No produce modelo interpretable

Estandarización es crucial

Si X_1 está en [0, 100] y X_2 en [0, 1], la distancia estará dominada por X_1. **Siempre estandarizar** antes de KNN.

La Maldición de la Dimensionalidad

¿Por qué KNN sufre con muchas variables?

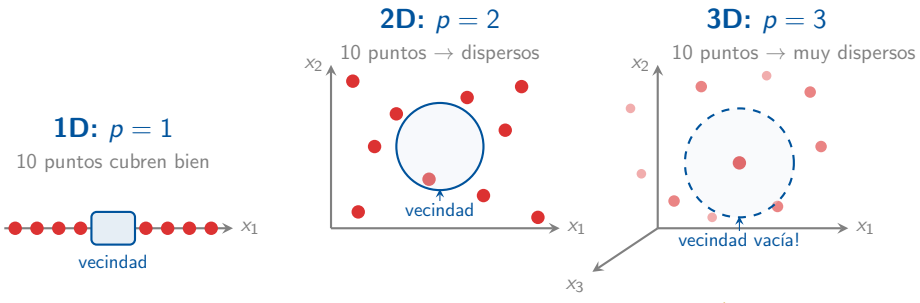
A medida que p crece, el espacio se vuelve exponencialmente vacío:

- En 1D: 10 puntos cubren bien el intervalo $[0, 1]$
- En 2D: Necesitas $10^2 = 100$ puntos para la misma densidad
- En 10D: Necesitas 10^{10} puntos
- En 100D: Necesitas 10^{100} puntos (más que átomos en el universo)

Consecuencia para KNN:

- Los "vecinos cercanos" ya no son cercanos
- Todos los puntos están aproximadamente a la misma distancia
- El voto mayoritario pierde sentido

La Maldición de la Dimensionalidad



Para cubrir el 10 % del rango en cada dimensión necesitas: $p = 1$: 10 % de los datos $p = 2$: $0{,}1^2 = 1\,\%$ $p = 10$: $0{,}1^{10} = 0{,}00000001\,\%$

Agenda de hoy

SVM: Motivación

El problema del banquero conservador:
Imagina que tienes que trazar una línea para separar:

- Clientes que pagarán (buenos)
- Clientes que harán default (malos)

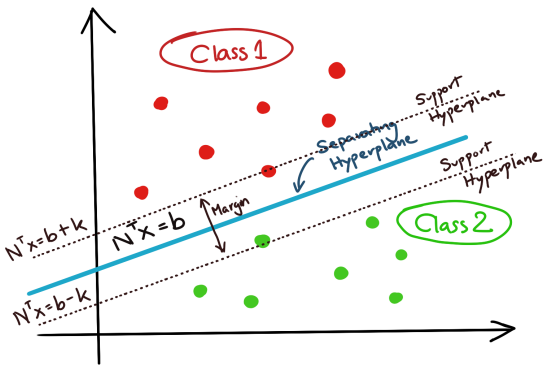
[TABLERO : Infinitos hiperplanos separadores]

Hay infinitos hiperplanos que separan las clases

Pregunta: ¿Cuál línea elegirías?
Respuesta de SVM: La más **conservadora** - la que se aleja lo máximo posible de ambos grupos (margen máximo).

Si llegan clientes nuevos ligeramente diferentes, la línea con mayor margen cometerá menos errores.

SVM: La Idea del Margen Máximo



Componentes visuales:

- **Línea central:** frontera de decisión
- **Líneas punteadas:** márgenes
- **Puntos clave:** vectores de soporte

Problema original (hard margin):

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2$$

sujeto a

$$y_i(w^\top x_i + b) \geq 1$$

Idea: separación perfecta maximizando el margen.

Vectores de Soporte: Los Puntos Críticos

¿Qué son los vectores de soporte?
Son los puntos de entrenamiento que están **exactamente en el margen** - los más difíciles de clasificar.

Intuición económica:
Imagina clasificar empresas en riesgosas"vs "seguras":

- Empresa con deuda = 90 % y baja rentabilidad → Claramente riesgosa
- Empresa con deuda = 10 % y alta rentabilidad → Claramente segura
- Empresa con deuda = 50 % y rentabilidad media → En la frontera!

Los vectores de soporte son esos casos fronterizos.

Propiedad importante

SVM solo necesita estos puntos fronterizos para hacer predicciones. Los puntos "obvios"(muy alejados de la frontera) no importan para el modelo final.

¿Cómo Clasifica SVM?

Proceso de clasificación:

1. Entrenar: Encuentra la mejor línea/frontera que separa las clases
   - Maximizando el margen
   - Usando solo los vectores de soporte
2. Predecir: Para un nuevo punto
   - ¿De qué lado de la línea cae?
   - Esa es su clase predicha

Ventaja del margen grande:

- Si un nuevo cliente es ligeramente diferente a los del training
- Con margen grande, es menos probable que cruce la frontera
- Más robusto a variaciones

SVM con Datos No Separables: Soft Margin

El mundo real es imperfecto

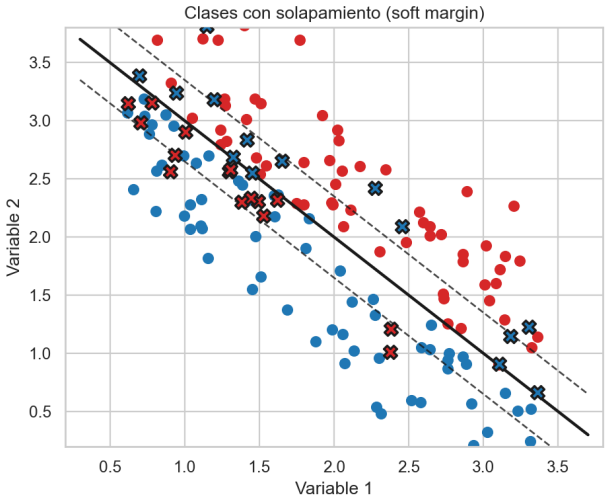
- Buenos clientes que hacen default
- Malos clientes que pagan

→ *Las clases reales se solapan*

Dilema del banquero:

- Margen estrecho → cero errores en training
- Margen amplio → mayor robustez

Soft Margin permite errores — pero los penaliza



Soft Margin: Permitir Algunos Errores

La realidad es desordenada:

En lugar de exigir separación perfecta, SVM permite:

- Puntos dentro del margen
- Algunos errores de clasificación

Pero los penaliza mediante variables de holgura.

Formulación Soft Margin SVM:

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \tag{19}$$

sujeto a:

$$y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \tag{20}$$

C controla el trade-off:

Hiperparámetro C en SVM

¿Qué controla C?
Trade-off entre:

- Margen grande

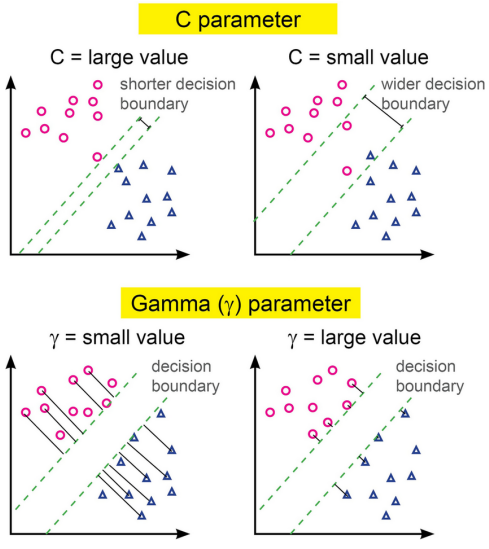
$$\Rightarrow \|w\|^2 \text{ pequeño}$$

- Errores de clasificación

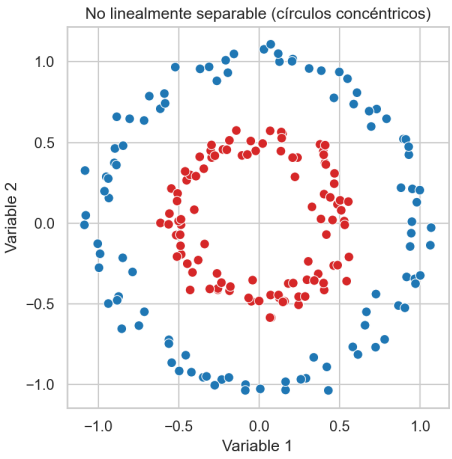
$$\Rightarrow \sum \xi_i \text{ pequeño}$$

Interpretación:

- C pequeño $\rightarrow$ margen amplio $\rightarrow$ underfitting
- C grande $\rightarrow$ margen estricto $\rightarrow$ overfitting



Kernel Trick: Magia de SVM



Problema del mundo real:
Las relaciones económicas rara vez son lineales.

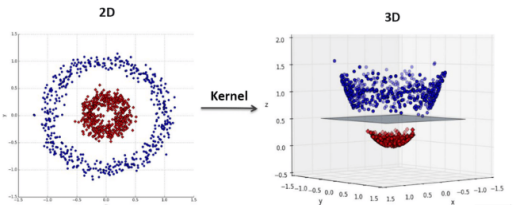
Ejemplo: riesgo de default

- Ingresos muy bajos $\rightarrow$ alto riesgo
- Ingresos medios $\rightarrow$ bajo riesgo
- Ingresos altos + deuda $\rightarrow$ alto riesgo

Idea del kernel:
Transformar el problema a un espacio donde la separación se vuelve simple.

Es como encontrar la representación correcta de los datos.

Mapeo a Espacio de Características: Más Dimensiones



La magia: Agregamos una tercera dimensión (por ejemplo, $x_1^2 + x_2^2$)

- En 2D: Necesitas una curva para separar
- En 3D: ¡Ahora puedes usar un plano!

Es como cuando en econometría agregamos X^2 para capturar efectos no lineales, pero SVM lo hace automáticamente.

El Kernel Trick: Magia Computacional

El problema: Si agregamos muchas dimensiones (100, 1000, ¡infinitas!), calcular se vuelve imposible.

El truco del kernel:
No necesitamos calcular explícitamente las nuevas dimensiones. Solo necesitamos saber qué tan "similares"son dos puntos en el nuevo espacio.

Intuición

En lugar de:

1. Transformar todos los datos a dimensión alta
2. Calcular distancias allá

Usamos una **función kernel** que calcula directamente:
"¿Qué tan similares son estos dos puntos en el espacio transformado?"
...sin hacer la transformación!

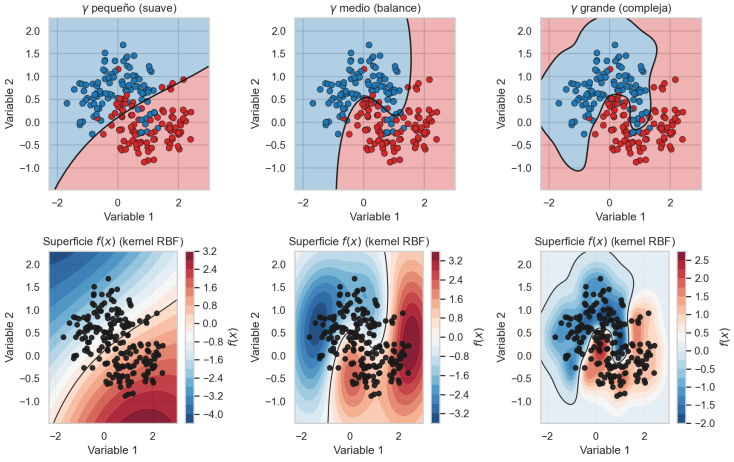
Kernels Populares: ¿Cuál Elegir?

Tipos principales de transformación:

1. **Kernel Lineal** - Sin transformación
   - Para cuando la relación ya es lineal
   - Más simple, más interpretable
2. **Kernel Polinomial** - Agrega potencias ($X^2, X^3, ...$)
   - Para relaciones de curvatura específica
   - Ejemplo: Efecto cuadrático del ingreso en default
3. **Kernel RBF (Gaussiano)** - El más popular
   - Funciona para casi cualquier patrón no lineal

Regla práctica: Empieza con RBF, si es muy lento prueba lineal.

Kernel RBF: El Más Popular



- γ **controla la influencia local:**
- γ **pequeño**
  - Influencia "lejos"
  - Frontera suave
  - Underfitting
- γ **grande**
  - Influencia "cerca"
  - Frontera compleja
  - Overfitting

Analogía: Promedios regionales vs locales.

Hiperparámetros en SVM: Lo Esencial

Lo que necesitas tunear:

1. C**:** ¿Qué tan estricto soy con los errores?
   - C pequeño: Margen amplio, tolero errores (simple)
   - C grande: Margen estrecho, castigo errores (complejo)

2. **Kernel:** ¿Qué tipo de relación espero?
   - Lineal: Relación lineal
   - RBF: No sé, probablemente no lineal (default)

3. γ **(solo si RBF):** ¿Qué tan local es la influencia?
   - γ pequeño: Influencia amplia (simple)
   - γ grande: Influencia local (complejo)

Grid Search + Cross-Validation para encontrar los mejores valores.

SVM: Ventajas y Desventajas

Ventajas:

- Funciona bien en alta dimensión (p grande)
- Robusto a overfitting (principio de margen máximo)
- Kernel trick permite fronteras complejas
- Fundamentación matemática sólida
- Solo depende de vectores de soporte

Desventajas:

- Lento de entrenar con n muy grande
- Requiere tuning cuidadoso de C y γ
- Difícil de interpretar (caja negra)
- No produce probabilidades naturalmente
- Sensible a escalas (requiere estandarización)

Agenda de hoy

KNN vs Logit vs SVM: Comparación

	KNN	**Logit**	**SVM**
Tipo	No paramétrico	Paramétrico	No paramétrico
Frontera	Puede ser compleja	Lineal	Lineal/No lineal
Entrenamiento	Ninguno	Rápido	Lento (n grande)
Predicción	Lento	Rápido	Rápido
Alta dim. (p)	Mal	Bien	Muy bien
Interpretabilidad	Nula	Alta	Baja
Probabilidades	Sí	Sí (nativo)	No (requiere calibración)
Hiperparáms.	K	λ (si reg.)	C, kernel, γ

¿Cuándo Usar Cada Uno?

Regresión Logística:

- Primera opción para clasificación binaria
- Cuando se necesita interpretabilidad
- Cuando se necesitan probabilidades calibradas
- Baseline para comparar otros modelos

KNN:

- Datasets pequeños ($n < 10{,}000$)
- Baja dimensionalidad ($p < 20$)
- Como baseline rápido
- Cuando la frontera es muy irregular localmente

SVM:

- Cuando p es grande (alta dimensión)
- Fronteras de decisión complejas
- Cuando accuracy es más importante que interpretabilidad
- Datos no linealmente separables (con kernel RBF)

Resumen: Ideas Clave

1. **Regresión Logística:** El puente desde econometría
   - Interpreta con odds ratios y efectos marginales
2. **KNN:** "Dime con quién andas..."
   - Simple pero sensible a dimensionalidad
   - Hiperparámetro K controla flexibilidad
3. **SVM:** El banquero conservador
   - Maximiza margen para robustez
   - Kernel trick permite no linealidad
   - Hiperparámetros: C (tolerancia) y γ (localidad)

Próxima clase: Métricas de clasificación - más allá del accuracy.

¡Gracias!

s.neira10@uniandes.edu.co