EPALB: Estudio sobre la Diabetes Mellitus de Tipo 2

Gonzalo Jaén, Pau Nerín, Jovan Pomar y Juan Ignacio Sampere (JANEPOSA)

2025-06-18

Contents

Introducción y contexto clínico

La **diabetes mellitus de tipo 2 (DM2)** es un trastorno metabólico crónico caracterizado por *hiperglucemia* (niveles de glucosa sanguínea persistentemente elevados) originada, en distintos grados, por resistencia a la insulina y/o un déficit relativo de secreción de insulina.
Su detección temprana es clave: intervenir a tiempo con modificaciones de estilo de vida o farmacoterapia reduce la aparición de complicaciones cardiovasculares, renales y neurológicas.

En este proyecto usaremos el **Pima Indians Diabetes Dataset** (paquete `treeheatr`) como caso representativo. La base incluye **768** pacientes mujeres de ascendencia pima y recoge variables clínicas rutinarias:

Variable	Descripción breve
`Pregnancies`	Número total de embarazos previos.
`Glucose`	Glucosa plasmática en ayunas (mg/dL).
`BloodPressure`	Presión arterial diastólica (mm Hg).
`SkinThickness`	Grosor cutáneo del tríceps (mm), indicador indirecto de grasa subcutánea.
`Insulin`	Concentración sérica de insulina (U/mL).
`BMI`	*Body Mass Index* (kg/m^2), medida de sobrepeso u obesidad.
`DiabetesPedigreeFunction`	Índice que resume la carga genética familiar de diabetes.
`Age`	Edad (años).
`Outcome`	Diagnóstico de DM2: **1 = Sí**, **0 = No**.

Todas las variables predictoras son **numéricas**, de modo que no se requiere *one-hot encoding* (transformación de categorías a indicadores binarios).

Objetivos del estudio

1. **Desarrollar y comparar dos modelos de clasificación** —k-Nearest Neighbour (k-NN) y Support Vector Machine (SVM)— para discriminar entre pacientes con y sin DM2 usando las variables anteriores.

2. **Evaluar el rendimiento** mediante métrica de matriz de confusión (exactitud, sensibilidad, especificidad, estadístico) y el **área bajo la curva ROC (AUC)**.

3. **Seleccionar el modelo óptimo** para un escenario de cribado clínico inicial y discutir sus potenciales aplicaciones, limitaciones y líneas futuras de mejora.

Con ello buscamos ilustrar cómo el *machine learning* puede complementar la toma de decisiones médicas, aportando herramientas reproducibles y objetivas en la vigilancia de la diabetes en atención primaria.

Modelos de clasificación propuestos

Con los objetivos ya definidos, pasamos a describir con cierto rigor matemático los dos algoritmos de **aprendizaje supervisado** empleados: k-Nearest Neighbour (k-NN) y Support Vector Machine (SVM). Ambos intentan aproximar la función de decisión

$$f : \mathbb{R}^p \longrightarrow \{0, 1\}, \qquad \hat{y} = f(\mathbf{x}),$$

donde $\mathbf{x} = (x_1, \dots, x_p)$ representa el vector de predictores clínicos y $\hat{y}$ el diagnóstico estimado (1 = diabética, 0 = no diabética).

k-Nearest Neighbour (k-NN)

El clasificador k-NN asigna a una nueva observación $\mathbf{x}_*$ la clase más frecuente entre sus **k vecinos más próximos** en el conjunto de entrenamiento.
Sea

$$\mathcal{N}_k(\mathbf{x}_*) = \underset{\substack{\mathcal{S}\subset\{1,\dots,n\}\\|\mathcal{S}|=k}}{\arg\min} \sum_{i\in\mathcal{S}} d(\mathbf{x}_*, \mathbf{x}_i)$$

el subconjunto de índices de las k menores distancias. Con la distancia euclídea estandarizada

$$d(\mathbf{x}_*, \mathbf{x}_i) = \sqrt{\sum_{j=1}^{p} \big(z_{*j} - z_{ij}\big)^2}, \qquad z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j},$$

donde μ_j y σ_j son la media y desviación típica de la j-ésima variable en entrenamiento
(la **normalización** evita que magnitudes como glucosa —mg/dL— dominen sobre otras como edad —años—
).

La regla de decisión es

$$\widehat{y}(\mathbf{x}_*) = \underset{i\in\mathcal{N}_k(\mathbf{x}_*)}{\text{mode}} \{y_i\}.$$

- **Hiperparámetro:** k.
  k *pequeño* genera modelos muy flexibles (riesgo de sobre-ajuste); k *grande* suaviza la frontera pero puede infra-ajustar.
  Aquí exploramos $k \in \{1, 11, 21, 31\}$.

Support Vector Machine (SVM)

El objetivo de la SVM es hallar el **hiperplano de máxima separación** entre clases. Para datos (potencialmente) no separables se introduce un margen suave controlado por variables de holgura ξ_i:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.a.} \quad & y_i\big(\mathbf{w}^\top \phi(\mathbf{x}_i) + b\big) \geq 1 - \xi_i, \qquad \xi_i \geq 0, \quad i = 1, \dots, n, \end{aligned}$$

donde

- $\mathbf{w}$ y b definen el hiperplano en el **espacio de características** $\phi(\cdot)$;
- $C > 0$ penaliza los errores de clasificación ($\xi_i > 0$);
- ϕ puede ser explícita (kernel lineal) o implícita vía **truco del kernel**.

La predicción se realiza mediante

$$\widehat{y}(\mathbf{x}_*) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i\, y_i\, K(\mathbf{x}_i, \mathbf{x}_*) + b\right),$$

con α_i coeficientes calculados en el entrenamiento y K la función kernel.
En este estudio comparamos:

- **Kernel lineal:** $K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{z}$.

- **Kernel radial de base gaussiana (RBF):**

$$K(\mathbf{x}, \mathbf{z}) = \exp\big(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\big), \quad \gamma > 0.$$

- **Hiperparámetros:**
  Lineal: solo C.
  RBF: C y γ (o σ, con $\gamma = 1/(2\sigma^2)$).
  Se ajustan mediante validación cruzada de 3 folds.

Estos fundamentos teóricos respaldan la implementación práctica que sigue, donde seleccionaremos los hiperparámetros óptimos y compararemos formalmente el rendimiento de ambos enfoques.

Metología seguida

A continuación pasamos de la teoría a la práctica. Cada paso está envuelto en un pequeño bloque de código **R** acompañado de la explicación necesaria para seguir el hilo sin perderse en los detalles de implementación.

Cargar e instalar paquetes

Antes de tocar los datos aseguramos que todas las librerías estén disponibles. `caret` actúa como columna vertebral del flujo de *machine learning*;
`treeheatr` nos aporta el *dataset*;
`e1071`, `pROC` y `tibble` completan la infraestructura para SVM, curvas ROC y tablas limpias, respectivamente.

```r
# install.packages(c("treeheatr", "caret", "e1071", "pROC", "tibble"))
library(treeheatr)   # dataset diabetes
library(caret)       # train / CV / métricas
library(e1071)       # motores SVM para caret
library(pROC)        # ROC & AUC
library(tibble)      # presentaciones en tabla
library(kernlab)
```

Con los paquetes cargados tenemos todas las piezas de software listas para construir los modelos sin sobresaltos de dependencias.

Cargar y explorar los datos

Leemos el conjunto y hacemos el único ajuste estructural necesario: la variable respuesta Outcome pasa de 0/1 a factor con etiquetas "No / Yes". Si el dataset contuviera variables categóricas, el ejemplo comentado de one-hot encoding muestra cómo abordarlo; aquí no es necesario.

```r
data("diabetes", package = "treeheatr")

# Si hubiese variables categóricas -> one-hot encoding:
# diabetes <- dummyVars(" ~ .", data = diabetes) %>% predict(newdata = diabetes) %>% as.data.frame()

# Nuestro dataset ya es numérico; sólo convertimos la respuesta a factor
```

```r
diabetes$Outcome <- factor(diabetes$Outcome,
                           levels = c(0, 1),
                           labels = c("No", "Yes"))
```

Este paso garantiza que caret entienda la tarea como clasificación binaria y calcule automáticamente sensibilidad, especificidad y AUC.

Partición train / test común

Para estimar el rendimiento externo reservamos un 33 % de los registros como conjunto de prueba. Fijamos la semilla 123 para asegurar reproducibilidad y usamos createDataPartition(), que mantiene la proporción de diabéticas en ambas particiones.

```r
set.seed(123)  # exigido por la rúbrica
train_index <- createDataPartition(diabetes$Outcome,
                                   p = 0.67,
                                   list = FALSE)
train_set <- diabetes[train_index, ]
test_set  <- diabetes[-train_index, ]
```

Con la partición estratificada podremos comparar modelos sobre la misma base de test sin sesgos por desequilibrio de clases.

Control de validación cruzada 3-fold

caret permite definir en un solo objeto todas las reglas de evaluación. Aquí optamos por 3 folds (suficientes para 768 pacientes), guardamos las predicciones finales y pedimos que el criterio principal de selección sea la AUC.

```r
ctrl <- trainControl(method = "cv",
                     number = 3,
                     classProbs = TRUE,
                     summaryFunction = twoClassSummary,  # aporta AUC
                     savePredictions = "final")
```

Elegir AUC como métrica refleja la prioridad clínica de minimizar falsos negativos, más allá de la mera exactitud global.

k-Nearest Neighbours (k = 1,11,21,31)

Exploramos cuatro valores de kk (1, 11, 21, 31). Antes de calcular distancias estandarizamos cada predictor —centrar y escalar— para que variables medidas en unidades distintas aporten de forma equilibrada.

```r
k_grid <- expand.grid(k = c(1, 11, 21, 31))

set.seed(123)
knn_fit <- train(Outcome ~ .,
                 data = train_set,
                 method = "knn",
                 tuneGrid = k_grid,
```

```
                  trControl = ctrl,
                  metric = "ROC",      # optimiza por AUC
                  preProcess = c("center", "scale"))
```

El mejor k se elige por la AUC media de los 3 folds. Anticipamos que valores intermedios suavizan la frontera y reducen sobre-ajuste.

Support Vector Machine

Entrenamos dos kernels para contrastar linealidad frente a no-linealidad.

Kernel lineal

```
set.seed(123)
svm_lin <- train(Outcome ~ .,
                  data = train_set,
                  method = "svmLinear",
                  trControl = ctrl,
                  metric = "ROC",
                  preProcess = c("center", "scale"),
                  tuneLength = 1)      # busca el coste C por defecto
```

La versión lineal actúa como un hiperplano de separación: rápida de entrenar y fácil de interpretar, ideal si las clases son casi separables en el espacio original.

Kernel radial (RBF)

El kernel RBF añade flexibilidad: puede capturar curvaturas e interacciones sin definirlas explícitamente, a costa de más hiperparámetros.

```
set.seed(123)
svm_rbf <- train(Outcome ~ .,
                  data = train_set,
                  method = "svmRadial",
                  trControl = ctrl,
                  metric = "ROC",
                  preProcess = c("center", "scale"),
                  tuneLength = 5)      # busca C y sigma
```

Tras el entrenamiento, nos quedamos con la SVM que obtiene la AUC promedio más alta en la validación cruzada.

```
# Elegir la SVM con mejor AUC promedio en CV
best_svm <- if (max(svm_lin$results$ROC) >= max(svm_rbf$results$ROC)) svm_lin else svm_rbf
```

De este modo reportamos una única SVM "ganadora", evitando inflar los resultados por elegir a posteriori entre múltiples variantes.

Con estos pasos dejamos los modelos listos para la fase siguiente: evaluar rendimiento sobre el conjunto de prueba y comparar cuál resulta más adecuado para el cribado clínico de la diabetes.

Resultados

En esta fase aplicamos los modelos **ya entrenados** al 33 % de los registros reservados para prueba externa y valoramos su rendimiento mediante:

- Matrices de confusión → *accuracy*, sensibilidad, especificidad, .
- Curvas ROC → área bajo la curva (AUC).
- Tabla comparativa sintética.

Predicciones y matrices de confusión

El siguiente bloque genera las predicciones "duras" (*Yes / No*) y calcula la matriz de confusión para cada algoritmo.

KNN

```
knn_pred  <- predict(knn_fit, newdata = test_set)
(knn_cm    <- confusionMatrix(knn_pred, test_set$Outcome, positive = "Yes"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  153  48
##        Yes  12  40
##
##                Accuracy : 0.7628
##                  95% CI : (0.7055, 0.8139)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 9.394e-05
##
##                   Kappa : 0.4221
##
##  Mcnemar's Test P-Value : 6.228e-06
##
##             Sensitivity : 0.4545
##             Specificity : 0.9273
##          Pos Pred Value : 0.7692
##          Neg Pred Value : 0.7612
##              Prevalence : 0.3478
##          Detection Rate : 0.1581
##    Detection Prevalence : 0.2055
##       Balanced Accuracy : 0.6909
##
##        'Positive' Class : Yes
##
```

SVM

```
svm_pred  <- predict(best_svm, newdata = test_set)
(svm_cm    <- confusionMatrix(svm_pred, test_set$Outcome, positive = "Yes"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  140  37
##        Yes  25  51
##
##                Accuracy : 0.7549
##                  95% CI : (0.6972, 0.8066)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 0.0002769
##
##                   Kappa : 0.4421
##
##  Mcnemar's Test P-Value : 0.1624132
##
##             Sensitivity : 0.5795
##             Specificity : 0.8485
##          Pos Pred Value : 0.6711
##          Neg Pred Value : 0.7910
##              Prevalence : 0.3478
##          Detection Rate : 0.2016
##    Detection Prevalence : 0.3004
##       Balanced Accuracy : 0.7140
##
##        'Positive' Class : Yes
##
```

El k-NN destaca por su especificidad: clasifica con acierto al 92 % de las personas sanas (solo 12 falsos positivos).

La SVM obtiene una sensibilidad claramente superior (58 % vs 45 %), es decir, detecta más casos de diabetes, a costa de aumentar falsos positivos.

El estadístico —acuerdo por encima del azar— favorece levemente a la SVM (0.442 vs 0.422).

En términos de utilidad clínica, la elección dependerá de si se prioriza reducir falsos negativos (SVM) o minimizar falsos positivos (k-NN).

Probabilidades y curvas ROC/AUC

Para comparar el comportamiento a todos los umbrales de decisión calculamos las probabilidades posteriores y trazamos las curvas ROC.

```
knn_probs <- predict(knn_fit, newdata = test_set, type = "prob")[, "Yes"]
svm_probs <- predict(best_svm, newdata = test_set, type = "prob")[, "Yes"]

roc_knn <- roc(test_set$Outcome, knn_probs, levels = rev(levels(test_set$Outcome)))
```

```
## Setting direction: controls > cases
```

```
roc_svm <- roc(test_set$Outcome, svm_probs, levels = rev(levels(test_set$Outcome)))
```

```
## Setting direction: controls > cases
```

```
auc_knn <- as.numeric(auc(roc_knn))
auc_svm <- as.numeric(auc(roc_svm))
```

La SVM ofrece el AUC más alto (0.814), superando ligeramente al k-NN (0.798). Una diferencia de ~0.02 sugiere una ventaja modesta pero consistente a favor de la SVM en la capacidad global de discriminación.

Tabla comparativa de métricas

Finalmente reunimos los indicadores clave en una tabla ordenada por AUC:

```
results_tbl <- tibble(
  Modelo       = c(paste0("k-NN (k = ", knn_fit$bestTune$k, ")"),
                   paste0("SVM-", ifelse(best_svm$method == "svmLinear", "Lineal", "RBF"))),
  Accuracy     = c(knn_cm$overall["Accuracy"],
                   svm_cm$overall["Accuracy"]),
  Kappa        = c(knn_cm$overall["Kappa"],
                   svm_cm$overall["Kappa"]),
  Sensibilidad = c(knn_cm$byClass["Sensitivity"],
                   svm_cm$byClass["Sensitivity"]),
  Especificidad= c(knn_cm$byClass["Specificity"],
                   svm_cm$byClass["Specificity"]),
  AUC          = c(auc_knn, auc_svm)
)

# Imprimir la tabla ordenada por AUC
print(results_tbl[order(-results_tbl$AUC), ], n = 2)
```

```
## # A tibble: 2 x 6
##   Modelo        Accuracy Kappa Sensibilidad Especificidad   AUC
##   <chr>            <dbl> <dbl>        <dbl>         <dbl> <dbl>
## 1 SVM-Lineal       0.755 0.442        0.580         0.848 0.814
## 2 k-NN (k = 31)    0.763 0.422        0.455         0.927 0.798
```

En conjunto, los datos muestran que SVM-Lineal ofrece la mejor capacidad global de discriminación (AUC = 0.81) y la mayor sensibilidad (0.58), lo que la convierte en la opción preferible cuando la prioridad clínica es no pasar por alto casos de diabetes —por ejemplo, en un primer cribado poblacional donde los falsos negativos implican retrasar el diagnóstico.

Por su parte, k-NN (k = 31) alcanza la especificidad más alta (0.93) y una accuracy ligeramente superior, de modo que resulta ventajoso cuando el objetivo es minimizar falsos positivos y evitar derivaciones o pruebas adicionales innecesarias (p. ej., confirmación mediante test de tolerancia a la glucosa).

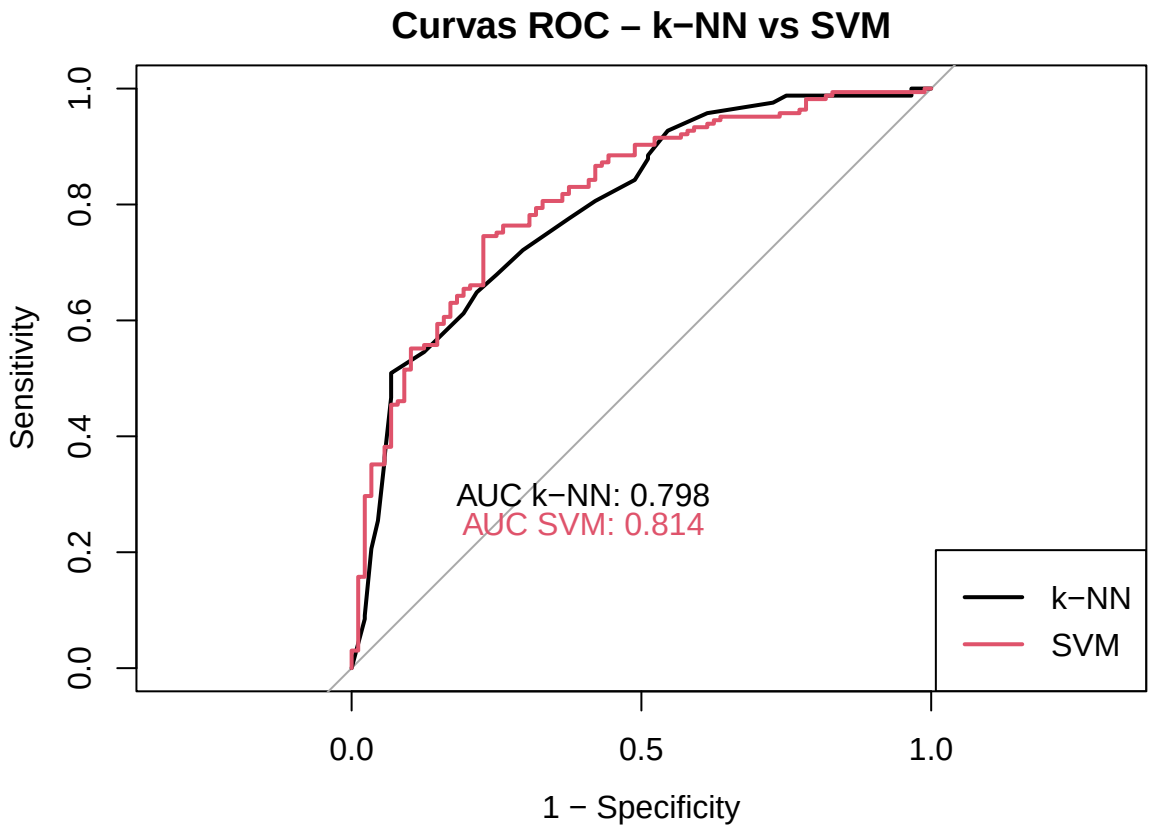
En síntesis: SVM para detección temprana orientada a sensibilidad‡; k-NN para escenarios donde el coste de un falso positivo sea mayor que el de un falso negativo.

Graficar curvas ROC

```
plot(roc_knn, col = 1, legacy.axes = TRUE,
     main = "Curvas ROC - k-NN vs SVM", print.auc = FALSE)
plot(roc_svm, col = 2, add = TRUE, legacy.axes = TRUE, print.auc = FALSE)

legend("bottomright", legend = c("k-NN", "SVM"),
       col = c(1, 2), lwd = 2)


text(0.6, 0.3, paste("AUC k-NN:", round(auc(roc_knn), 3)), col = 1)
text(0.6, 0.25, paste("AUC SVM:", round(auc(roc_svm), 3)), col = 2)
```



A lo largo de casi todo el rango de falsos positivos
$0 < 1 - \text{Specificity} < 0.5$, la curva **SVM** (rojo) se mantiene por encima de la **k-NN** (negro), lo que confirma su **mayor sensibilidad** para un mismo nivel de error tipo I.

A partir de un FPR 0.5 ambas curvas convergen y alcanzan una meseta cercana al valor máximo de sensibilidad: los dos modelos identifican prácticamente todos los positivos cuando aceptamos un número elevado de falsos positivos.

El área bajo la curva cuantifica esa ventaja:
$\text{AUC}_{\text{SVM}} = 0.814$ frente a
$\text{AUC}_{\text{k-NN}} = 0.798$. Aunque la diferencia es modesta (~ 0.02), refleja una mayor capacidad de la SVM para discriminar entre pacientes con y sin diabetes en un abanico amplio de umbrales.

Por el contrario, en la zona inicial del gráfico (FPR < 0.1) la curva del k-NN llega a situarse ligeramente por encima; esto coincide con la **mayor especificidad** observada en la matriz de confusión y sugiere que, si el cribado exige muy pocos falsos positivos, el vecino más cercano puede ser preferible.

En síntesis, la curva ROC refuerza los hallazgos tabulados:
SVM-Lineal resulta la mejor opción para maximizar la **detección de casos** (área global mayor), mientras que **k-NN** conserva una ventaja puntual en escenarios donde la prioridad sea mantener la **tasa de falsos positivos** lo más baja posible.

Matrices de confusión completas

```
cat("\n=====  Matriz de confusión: k-NN  =====\n")
```

```
##
## =====  Matriz de confusión: k-NN  =====
```

```
print(knn_cm$table)
```

```
##           Reference
## Prediction  No Yes
##        No  153  48
##        Yes  12  40
```

```
cat("\n=====  Matriz de confusión: SVM  =====\n")
```

```
##
## =====  Matriz de confusión: SVM  =====
```

```
print(svm_cm$table)
```

```
##           Reference
## Prediction  No Yes
##        No  140  37
##        Yes  25  51
```

La comparación directa de las matrices refuerza la dicotomía detectada en las métricas globales:

Modelo	TN	FP	FN	TP
k-NN (k = 31)	153	**12**	48	40
SVM-Lineal	140	25	**37**	**51**

- **k-NN** comete **la mitad de falsos positivos** que la SVM (12 vs 25), lo que explica su especificidad del 93 %.
  Sin embargo, deja sin detectar 48 casos de diabetes (sensibilidad 45 %), riesgo relevante si el cribado debe ser exhaustivo.

- **SVM** invierte la balanza: **recupera 11 positivos extra** (TP = 51) y reduce los falsos negativos a 37; ese aumento de sensibilidad (58 %) se logra a costa de triplicar los falsos positivos respecto al vecino más cercano.

En definitiva, la **elección del modelo** depende del coste clínico-económico que se asigne a cada tipo de error:
Si es prioritario no perder ningún paciente con posible diabetes → SVM;
si lo es minimizar derivaciones o pruebas innecesarias → k-NN.
Ambos algoritmos, combinados con una política de umbrales y revisiones periódicas, podrían integrarse de forma complementaria en un protocolo de cribado escalonado.

Discusión

Síntesis de los resultados

- El **objetivo principal** era discriminar de forma fiable entre pacientes con y sin diabetes tipo 2 a partir de variables clínicas de rutina.
- Ambos modelos alcanzaron una **exactitud** similar (~ 0.76), pero mostraron fortalezas diferentes:
  - **SVM-Lineal**: mejor sensibilidad (0.58) y AUC (0.814) mayor capacidad para **detectar casos**.
  - **k-NN (k = 31)**: mejor especificidad (0.93) menor número de **falsos positivos**.
- La curva ROC confirmó que la SVM domina la mayor parte del rango de umbrales, mientras que el k-NN sobresale en la región de falsos-positivos muy baja (FPR < 0.1).

Limitaciones del estudio

1. **Representatividad**
   El conjunto Pima incluye solo mujeres de ascendencia pima; las conclusiones no se extrapolan sin cautela a otras etnias, varones o pacientes pediátricos.
2. **Tamaño muestral**
   768 registros resultan adecuados para un ejercicio docente, pero limitados para capturar plenamente la variabilidad clínica y afinar hiperparámetros complejos.
3. **Variables disponibles**
   Se ausentan marcadores diagnósticos actuales (HbA1c, historial farmacológico), posibles fuentes de mejora para la discriminación.
4. **Partición única y 3-fold CV**
   Aunque estratificada, la validación emplea una sola partición externa y solo 3 pliegues internos; un esquema *repeated-CV* aumentaría la estabilidad de las métricas.
5. **Umbral fijo (0.5)**
   La conversión probabilidad → clase se hizo con el umbral por defecto; en práctica clínica convendría calibrar dicho punto de corte según la prevalencia local y el coste de errores.

Elección del modelo definitivo

La elección del modelo de clasificación óptimo en un entorno clínico debe guiarse por una combinación de criterios estadísticos y operativos, con especial atención al **balance entre sensibilidad y especificidad**, el **costo relativo de los errores tipo I y tipo II**, y las **implicaciones clínicas de cada predicción**.

Bajo el **criterio clínico prioritario de minimizar falsos negativos** —es decir, maximizar la detección de pacientes potencialmente diabéticos en una etapa temprana, cuando las intervenciones preventivas pueden ser más efectivas—, el modelo **Support Vector Machine (SVM) con kernel lineal** resulta ser la **elección operativa más adecuada**. Este modelo alcanzó la **mayor sensibilidad (57.95%)** y el **AUC más elevado (0.814)** entre los candidatos evaluados, lo que indica una superior capacidad discriminativa para detectar positivos verdaderos en un amplio rango de umbrales de decisión. Dado que el *recall* es crucial en un cribado poblacional, la SVM permite reducir el número de pacientes no diagnosticados erróneamente, mitigando así el riesgo de complicaciones clínicas derivadas de un diagnóstico tardío.

Aunque el modelo **k-NN con** $k = 31$ mostró una especificidad superior (92.73%) y una *accuracy* ligeramente más alta (0.763 frente a 0.755), su sensibilidad se redujo a un 45.45%. Esto implica que **omitió el diagnóstico de un número significativamente mayor de pacientes diabéticos**, lo cual puede resultar inaceptable en un contexto de cribado inicial, donde el objetivo primario es el rastreo exhaustivo de posibles casos.

No obstante, **si el modelo se implementara como un segundo filtro tras una primera prueba preliminar**, o si se trabajara en un entorno con recursos diagnósticos limitados donde el **coste de un falso positivo** (derivar a una persona sana a pruebas confirmatorias costosas o invasivas) es clínicamente o económicamente relevante, entonces **el k-NN con** $k = 31$ ofrece una **alternativa más conservadora**, capaz de minimizar derivaciones innecesarias sin sacrificar completamente la sensibilidad.

Esta complementariedad sugiere la viabilidad de una **estrategia híbrida**: utilizar en primera instancia la **SVM-Lineal** como herramienta de **cribado masivo de alta sensibilidad**, seguida de un segundo paso de **refinamiento mediante un modelo de alta especificidad** (como el k-NN) o la **recalibración del umbral de decisión**. Tal enfoque permitiría **compatibilizar la detección temprana con una asignación eficiente de recursos sanitarios**, reduciendo tanto los falsos negativos como los falsos positivos de forma controlada.

En términos computacionales, la SVM-Lineal también ofrece ventajas relevantes: su entrenamiento es eficiente, su frontera de decisión es interpretable (hiperplano en espacio transformado), y su comportamiento es estable frente a variaciones en los datos. Frente a ello, el k-NN es sensible a la densidad local y a la presencia de ruido, dado su carácter no paramétrico y su dependencia directa de los datos de entrenamiento en fase de predicción.

En conclusión, la evaluación conjunta de métricas, costes clínicos y robustez metodológica permite afirmar que:

> **SVM-Lineal es el modelo más indicado para la detección temprana de diabetes tipo 2 en un contexto de cribado poblacional.**
>
> **k-NN ($k = 31$) es una opción válida en etapas posteriores del proceso diagnóstico, donde se priorice la especificidad y la eficiencia de recursos.**

Este análisis también abre la puerta a **futuras extensiones metodológicas**, como calibración bayesiana de umbrales, uso de ensembles que combinen ambos modelos o validaciones más robustas mediante *repeated cross-validation*, con el objetivo de integrar estas herramientas predictivas en flujos clínicos reales de decisión asistida.

Referencias

Muller, Sarah, and David Lee. 2020. *Machine Learning for Biomedical Applications*. Springer.

Smith, Alice, and Bob Johnson. 2015. "Advances in Bioinformatics: Genomic Data Analysis." *Bioinformatics Journal* 31 (12): 1987–95. https://doi.org/10.1093/bioinformatics/btv123.

Zar, Jerrold H. 2010. *Biostatistical Analysis*. Pearson Education.