Universidad Nacional Autónoma de México

Facultad de Ciencias

Aplicación del modelo de máquinas de soporte vectorial en mediciones de registros vocales para el diagnóstico de la enfermedad de Parkinson

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Actuaria

PRESENTA:

Karen Lizbeth Vidaur Rodríguez

TUTORA

Dra. Lizbeth Naranjo Albarrán

Ciudad Universitaria, Cd.Mx., 2018

1. Datos del alumno
Vidaur
Rodríguez
Karen Lizbeth
84635071
Universidad Nacional Autónoma de México
Facultad de Ciencias
Actuaría
310153353

2. Datos del asesor
Naranjo
Albarrán
Lizbeth

3. Datos de la tesis
Aplicación del modelo de máquinas de soporte vectorial en mediciones de registros vocales para el diagnóstico de la enfermedad de Parkinson
119 p.
2018

4. Palabras clave:
Aprendizaje automático, máquinas de soporte vectorial, núcleos.

Agradecimientos

Quiero mostrar mi enorme agradecimiento a mi asesora la Dra. Lizbeth Naranjo Albarrán por su confianza, paciencia y por brindarme su apoyo durante la realización de este trabajo y hasta la culminación de éste.

Le agradezco a mis sinodales la Dra. Verónica E. Arriola Ríos, el Dr. Ricardo Ramírez Aldana, el M. en C. José S. Zamora Muñoz y el M. en C. Antonio Soriano Flores por sus valiosos comentarios, observaciones y el tiempo dedicado a este trabajo.

Gracias a mis hermanos por sus ánimos, compañía y apoyo durante este proceso.

Todo esto no hubiera sido posible sin el amparo incondicional y el cariño que me otorgaron mis padres. Gracias a mi mamá por su paciencia y apoyo, ya que de forma incomparable me ayudó en mis buenos y malos momentos durante este proceso. Gracias a mi papá ya que sin su ayuda tanto económica como personal no hubiera logrado concluir mi tesis. Nunca me alcanzarán las palabras para demostrarles todo mi aprecio y mi agradecimiento.

Índice general

Índice de figuras

Índice de tablas

Resumen

La enfermedad de Parkinson (EP) es una patología degenerativa caracterizada por sus síntomas motores y no motores, los cuales generan serias dificultades en las actividades cotidianas de las personas que padecen. Conforme progresa la enfermedad, la calidad de vida y autonomía de las personas con enfermedad de Parkinson se ven mermadas dados el incremento y la gravedad de los síntomas. Los más conocidos son los síntomas motores como el temblor, la rigidez y la lentitud. Por su parte, los menos conocidos son los síntomas no motores, como los trastornos del ánimo (apatía, depresión) y la conducta, problemas de sueño y estreñimiento, que condicionan mucho la calidad de vida.

Es por ello que se realizó este estudio en donde se obtuvieron registros vocales de 80 personas (40 con la EP y 40 sin la EP), con un total de 44 mediciones de distintas características acústicas para cada persona. El objetivo del estudio es que con estos datos podamos clasificar a las personas que tengan la enfermedad y las que no padecen la enfermedad, esto a través de un modelo de aprendizaje autómatico conocido como modelo de máquinas de soporte vectorial (MSV), y así poder hacer más eficiente la discriminación de los pacientes.

Se analizaron las características de la voz porque uno de los síntomas más habituales de la enfermedad de Parkinson es la alteración motora del habla que afecta del 60 % al 80 % de los pacientes, el habla en estos casos se caracteriza por tener una sonoridad e intensidad monótona, de bajo tono y pobremente prosódica, que tiende a desvanecerse al final de la fonación. El habla se produce en ataques lentos y pausas significativas para respirar entre palabras y sílabas, reduciéndose la fluidez verbal y el ritmo. En ocasiones también se produce repetición de sílabas, palabras o frases.

En el primer capítulo se establece el tema de máquinas de soporte vectorial, se

muestran algunas nociones preliminares para poder entender este modelo un poco más, además de los tipos de clasificación en los que se puede presentar el modelo de MSV: clasificación binaria y multiclase. En la clasificación binaria puede haber casos de datos linealmente separables y de datos no linealmente separables, en la clasificacIón multiclase hay distintos métodos para poder resolver este tipo que es aún más complejo, se muestran los más usados comúnmente. En cada uno de estos casos se muestra un ejemplo para poder entender mejor lo que hace cada uno de ellos.

En el segundo capítulo se muestra una introducción sobre la enfermedad de Parkinson, así como la información detallada sobre los materiales, métodos usados, participantes y una explicación de cada variable contenida en la base de datos. El método aplicado de MSV se realizó con el software R, se elaboraron dos tipos de modelos: un modelo con todas las variables acústicas y otro modelo con tan solo 11 variables. En los dos modelos se hace prácticamente lo mismo, a diferencia claro del número de variables, se aplica el modelo de MSV probando distintos núcleos así como la predicción para saber qué tan eficiente es cada modelo.

En el tercer capítulo se muestran las conclusiones obtenidas después de haber realizado este trabajo.

Finalmente se muestra un anexo con el código del software R de la aplicación de este trabajo.

Capítulo 1

Preliminares

En este capítulo, cada sección fue estudiada de distintas fuentes. La sección del modelo de aprendizaje fue consultada del libro (Vapnik, 1998). Las secciones como la curva ROC y *k-fold cross validation* fueron consultadas del libro (James et al., 2013). La sección de Machine Learning fue consultada del libro (Barber, 2012).

1.1. Aprendizaje Automático

El aprendizaje automático (*Machine Learning*) es una de las áreas de más rápido crecimiento de la informática, con aplicaciones de gran alcance. El aprendizaje automático es el cuerpo de investigación relacionado con el análisis automatizado de datos a gran escala, es fundamental para muchas áreas de interés en informática y dominios de procesamiento de información a gran escala, los campos matemáticos de estadística, teoría de la información, teoría de juegos y optimización.

El Aprendizaje Automático es una disciplina científica y una rama del área de Inteligencia Artificial que crea sistemas que aprenden automáticamente, es decir, identifican patrones complejos en millones de datos. El objetivo de la máquina es poder aplicar a nuevos datos lo que se ha aprendido en el pasado y así poder predecir comportamientos futuros automáticamente. En este caso es poder aprender a discriminar correctamente a las personas que tienen la enfermedad de Parkinson de las que no lo presentan y así poder

ayudar a identificar su detección temprana.

En términos generales, los dos campos principales del aprendizaje automático son el aprendizaje supervisado y el aprendizaje no supervisado. En el aprendizaje supervisado, la atención se centra en la predicción precisa, mientras que en el aprendizaje no supervisado el objetivo es encontrar descripciones compactas de los datos. En este sentido, se distingue entre los datos que se utilizan para entrenar un modelo y los datos que se utilizan para probar el rendimiento del modelo entrenado.

- Aprendizaje supervisado.

  Dado un conjunto de datos $D = \{(x_n, y_n), n = 1, \ldots, N\}$, la tarea consiste en conocer la relación entre la entrada x y la salida y de manera que, cuando se le presente una entrada nueva x^* la salida pronosticada y^* sea precisa. El par (x^*, y^*) no está en D pero se supone que fue generado por el mismo proceso desconocido que generó D. En este tipo de aprendizaje nuestro interés es describir y conocer a y en términos de x. Desde una perspectiva probabilística, nos referimos principalmente a la distribución condicional $p(y|x, D)$. El término *supervisado* indica que hay un *supervisor* nocional que especifica la salida y para cada entrada x en los datos disponibles D. La salida también se denomina *etiqueta*, particularmente cuando se trata de la clasificación.

- Aprendizaje no supervisado

  Dado un conjunto de datos $D = \{x_n, n = 1, \ldots, N\}$ en el aprendizaje no supervisado, intentamos encontrar una descripción aceptable compacta de los datos. En el aprendizaje no supervisado no existe una variable de predicción especial, de modo que, desde una perspectiva probabilística, nos interesa modelar la distribución $p(x)$. La probabilidad de que el modelo genere los datos es una medida de la precisión de la descripción.

1.2. Modelo del aprendizaje

El modelo del aprendizaje contiene tres elementos:

- El generador de datos (ejemplos), **G**.
- El operador objetivo (supervisor), **S**.

- La máquina de aprendizaje, **LM**.

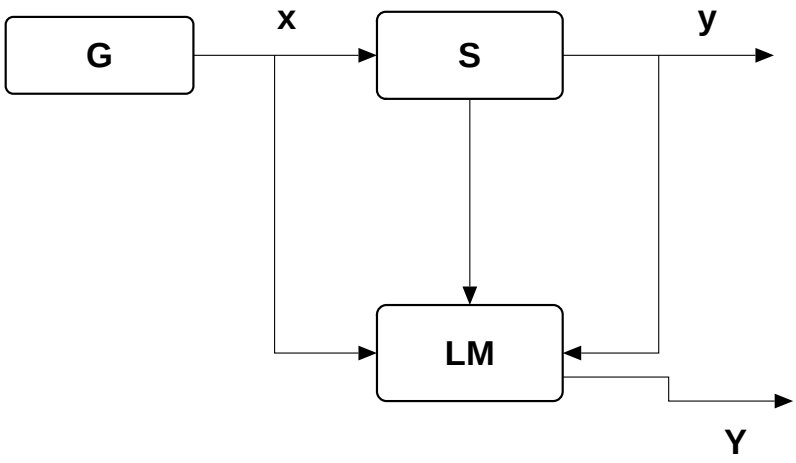


Figura 1.1: Diagrama del proceso de un modelo de aprendizaje.

La Figura 1.1 muestra el diagrama que ejemplifica el proceso de un modelo de aprendizaje.

La máquina de aprendizaje observa n pares de datos:

$$(x_1, y_1), \ldots, (x_n, y_n)$$

los cuales son denominados conjunto de entrenamiento, que contienen los vectores de entrada x y los vectores de respuesta y. Durante este periodo la máquina de aprendizaje elabora un operador que se utilizará para la predicción de la respuesta y; el objeto de la máquina de aprendizaje es elaborar una aproximación apropiada.

1.3. Curva ROC

La curva ROC es un gráfico que se utiliza para mostrar simultáneamente los dos tipos de errores (tasa de verdaderos positivos y tasa de falsos positivos) para todos los umbrales posibles. Sea D la variable binaria que denota el estado verdadero de la enfermedad, es decir:

$$D = \begin{cases} 1 & \text{para presencia de la enfermedad} \\ 0 & \text{para ausencia de la enfermedad} \end{cases} \tag{1.1}$$

y sea Y el resultado de la prueba:

$$Y = \begin{cases} 1 & \text{positivo para la enfermedad} \\ 0 & \text{negativo para la enfermedad} \end{cases} \tag{1.2}$$

los resultados de la prueba se resumen en las tablas 1.1 y 1.2.

	$D = 0$	$D = 1$
$Y = 0$	Especificidad o Razón de Verdaderos Negativos	Falsos Negativos (FN)
$Y = 1$	Falsos Positivos (FP)	Sensibilidad o Razón de Verdaderos Positivos

Tabla 1.1: Resultados: Curva ROC.

Probabilidades	Cálculo
Probabilidad de obtener un resultado negativo cuando el sujeto no tiene la enfermedad (especificidad)	$\frac{VN}{VN+FP}$
Probabilidad de obtener un resultado positivo cuando el sujeto tiene la enfermedad (sensibilidad)	$\frac{VP}{FN+VP}$
Proporción de resultados negativos que la prueba detecta como resultados positivos (Tasa de Falsos Positivos)	$\frac{FP}{FP+VN}$
Proporción de resultados válidos entre los resultados negativos de la prueba (valor predictivo negativo)	$\frac{VN}{VN+FN}$
Proporción de resultados válidos entre los resultados positivos de la prueba (valor predictivo positivo)	$\frac{VP}{VP+FP}$

Tabla 1.2: Resultados: Curva ROC.

donde VN = Verdaderos Negativos (número de resultados negativos que son realmente negativos), VP =Verdaderos Positivos (número de resultados positivos y que son verdaderamente positivos), FN =Falsos Negativos (número de resultados negativos y que en realidad son positivos) y FP =Falsos Positivos (número de resultados positivos y que en realidad son negativos).

La tasa verdadera positiva es la sensibilidad que es un parámetro que se mide en el grupo de sujetos que verdaderamente están enfermos. Es el cociente entre verdaderos positivos y el total de las personas enfermas. Por tanto, es la probabilidad de obtener un resultado positivo cuando el individuo tiene la enfermedad, utilizando un valor umbral dado. La tasa de falsos positivos es también conocida como 1-especificidad que es un parámetro que se mide en el grupo de sujetos no enfermos. Es el cociente entre verdaderos negativos y el total de no enfermos. Por tanto, es la probabilidad de obtener un resultado negativo cuando el individuo no tiene la enfermedad, utilizando ese mismo valor de umbral, ver figura 1.2.

Las curvas ROC son útiles para comparar diferentes clasificadores, ya que tienen en cuenta todos los umbrales posibles. La variación del umbral del clasificador cambia su verdadera tasa de positivos y falsos positivos. El nombre ROC (acrónimo de *Receiver Operating Characteristic* ó características de funcionamiento del receptor) es histórico y proviene de la teoría de las comunicaciones. El rendimiento general de un clasificador está dado por el área bajo la curva (AUC). Una curva ROC ideal se adherirá a la esquina superior izquierda lo que indica una alta tasa positiva verdadera y una baja tasa de falsos positivos, de modo que cuanto mayor sea el AUC, mejor será el clasificador. Explicaremos qué significan estos términos.

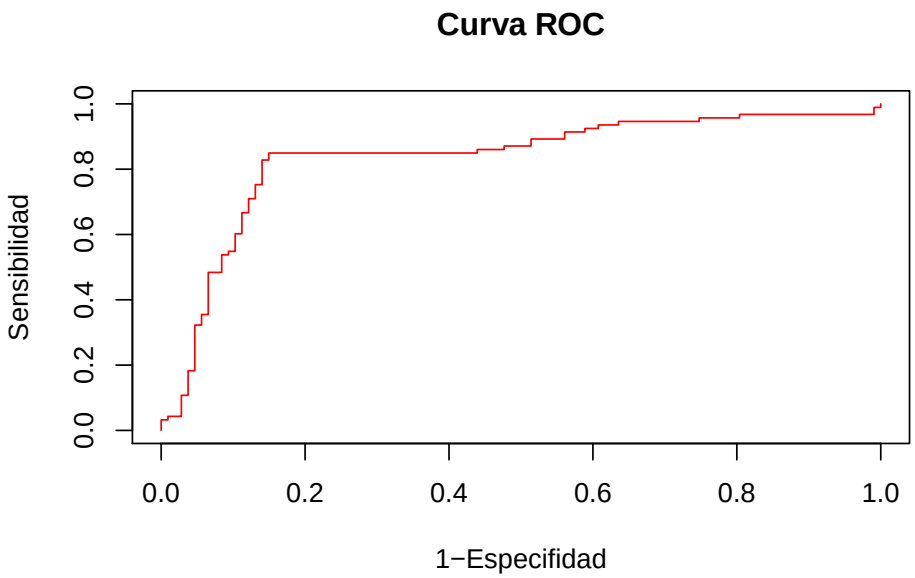


Figura 1.2: Curva ROC.

1.4. *k-fold cross validation* (validación cruzada k-veces)

Un método importante es la validación cruzada que sirve para estimar el MSE (*Mean Square Error* ó error cuadrático medio) utilizando los datos de entrenamiento (datos que entrenan al modelo, es decir es un conjunto de ejemplo que sirve para poder llegar al modelo final). Este método implica dividir aleatoriamente el conjunto de observaciones en k grupos de tamaño aproximadamente igual. El primer grupo se trata como un conjunto de validación, y el método es ajustado para los $k-1$ grupos restantes. El error cuadrático medio, MSE_i, se calcula en las observaciones en el grupo retenido. Este procedimiento se repite k veces; cada vez, un grupo diferente de observaciones se trata como un conjunto de validación. Este proceso da como resultado k estimaciones del error de prueba, MSE_1, MSE_2, ..., MSE_k. La estimación de la validación cruzada k-veces se calcula promediando estos

valores con la ecuación (1.3).

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i \tag{1.3}$$

En la práctica, uno normalmente realiza la validación cruzada k-veces usando $k = 5$ o $k = 10$. ¿Cuál es la ventaja de usar $k = 5$ o $k = 10$ en lugar de considerar las n observaciones ($k = n$)? La ventaja más obvia es computacional. La validación cruzada es un enfoque muy general que se puede aplicar a casi cualquier método de aprendizaje estadístico. Algunos métodos de aprendizaje estadístico tienen procedimientos de ajuste computacionalmente intensivos, por lo que realizar 10 veces la validación cruzada requiere ajustar el procedimiento de aprendizaje solo diez veces, lo que puede ser mucho más factible.

Capítulo 2

Máquinas de soporte vectorial

Las máquinas de soporte vectorial, constituyen una técnica recientemente desarrollada, efectúan una clasificación entre objetos puntuales de dos clases por medio de una superficie de decisión determinada por ciertos puntos del conjunto de entrenamiento, conocidos como vectores de soporte. Esta superficie se obtiene al resolver un problema de programación cuadrática (en casos binarios, ya que en casos multiclase se usan otro tipos de métodos que se mencionarán más adelante).

Para las secciones de máquinas de soporte vectorial para clasificación binaria, clasificador de margen máximo para datos linealmente separables, clasificador de margen suave y el caso no lineal se consultaron los libros (Shigeo, 2010),(Cristianini and Shawe-Taylor, 2000), (Hastie et al., 2001), (Kecman, 2005), (Osuna et al., 1997)y (James et al., 2013). En las secciones de la función núcleo y las máquinas de soporte vectorial para clasificación multiclase se consultó el libro (Shigeo, 2010). Los conceptos de dimensión VC y minimización del riesgo estructural fueron consultadas del libro (Vapnik, 1998).

2.1. Máquinas de soporte vectorial para la clasificación binaria

A continuación consideraremos funciones con restricciones lineales, (es decir que las funciones de separación serán hiperplanos) para la clasificación binaria de datos linealmente separables. En este caso, es posible clasificar

correctamente, lo que significa que el riesgo empírico puede ser igual a cero y que es un problema de clasificación muy sencillo. Generalmente este tipo de casos no se dan en la vida real es por ello que más adelante se estudiará el tema para datos linealmente no separables.

2.1.1. Clasificador de margen máximo para datos linealmente separables

Dado que el objetivo de las máquinas de soporte vectorial es encontrar el hiperplano que separe correctamente a las clases, habrá un número infinito de tales hiperplanos, es por ello que debemos elegir el hiperplano de margen máximo que es el que está más alejado de las observaciones de entrenamiento. En tal caso, podemos calcular la distancia perpendicular de cada observación de entrenamiento a un hiperplano de separación dado; el que tenga la menor distancia sería el hiperplano con el margen máximo y es el que posee el margen mayor. A esto se le conoce como hiperplano de separación óptimo. Cabe mencionar que esto sólo puede realizarse cuando los datos son linealmente separables.

Deseamos construir una MSV que sepa mapear los valores de los vectores x_i a los valores de las etiquetas y_i por medio de una función de decisión $y = f : \mathbb{R}^n \to \{1, -1\}$, la cual está entre dos hiperplanos que definen un margen máximo.

Sea $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, donde $x_i \epsilon \mathbb{R}^n, y_i \epsilon \{+1, -1\}$ un conjunto separable de n datos (para el caso de visualización sólo consideraremos el espacio de entrada bidimensional, es decir $x \epsilon R^2$) se puede definir un hiperplano de separación como una función lineal que es capaz de separar dicho conjunto sin error, entonces podemos determinar la función de decisión como:

$$D(x) = w^T x + b = \sum_{i=1}^{n} w_i x_i + b \tag{2.1}$$

donde $w, x \epsilon R^n$ (n es la dimensión del espacio de entrada) y b es el sesgo. Para ver cómo se clasifican los datos x_i se tiene la siguiente regla de decisión:

Si $D(x) > 0$, el dato x_i pertenece a la clase 1 ($y_i = +1$)
Si $D(x) < 0$, el dato x_i pertenece a la clase 2 ($y_i = -1$)

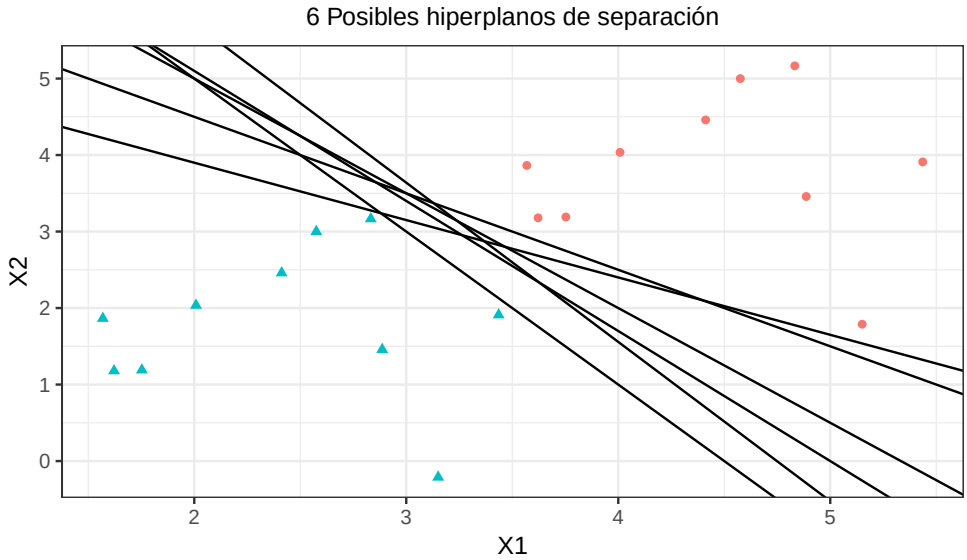


Figura 2.1: Posibles hiperplanos de separación en un conjunto linealmente separable.

Las desigualdades anteriores las deberá cumplir el hiperplano óptimo. Como vemos en la figura 2.1 hay una infinidad de hiperplanos que separan correctamente a las dos clases, y para determinar cuál es el óptimo lo que debemos hacer es maximizar el margen.

Debido a que los datos de entrenamiento son linealmente separables ningún dato de entrenamiento satisface $w^T x + b = 0$, por lo tanto para controlar la separabilidad en lugar de la regla de decisión anterior consideraremos las siguientes desigualdades:

$$w^T x + b \begin{cases} \geq 1 & \text{para } y_i = +1 \\ \leq 1 & \text{para } y_i = -1 \end{cases} \tag{2.2}$$

Para combinar ambas desigualdades podemos notar que $w^T x + b \leq -1$ es igual a $-(w^T x + b) \geq 1$ esto es parecido a $w^T x + b \geq 1$ excepto porque el anterior está multiplicado por un -1. Entonces el lado izquierdo de la desigualdad está multiplicado en ambos casos por su propia clase, por lo tanto tenemos que

$$y_i(w^T x + b) \geq 1 \quad \text{para} \quad i = 1, ..., n. \tag{2.3}$$

Entonces:

$$D(x) = w^T x + b = c \quad \text{para} \quad -1 < c < 1 \tag{2.4}$$

forma un hiperplano que separa a los patrones x_i. Cuando $c = 0$ está en medio de los dos hiperplanos $c = +1$ y -1 , entonces si

$$D(x) = 0 \tag{2.5}$$

el hiperplano tiene el margen máximo. El concepto de margen máximo está relacionado con la capacidad de generalización del hiperplano de separación, es decir que cuando se maximiza la capacidad de generalización entonces se está eligiendo el hiperplano de separación óptimo, así los datos que están en ambos lados de éste y que definen el margen, o lo que es lo mismo, aquellos que cumplen con la desigualdad (2.3), son los llamados *vectores de soporte.*

Decimos que el conjunto de vectores está óptimamente separado por el hiperplano si está separado sin error y la distancia entre el vector más cercano y el hiperplano es máxima, sin embargo, hay cierta redundancia en la ecuación (2.5), y sin pérdida de generalidad es apropiado considerar un hiperplano canónico donde los parámetros w, b están limitados por:

$$\underset{i}{\text{minimizar}} \quad |w^T x_i + b| = 1 \tag{2.6}$$

Estos puntos pueden pertenecer a cualquiera de las dos clases, particularmente si son de la clase 1 caen en el hiperplano $H_1 : w \cdot x + b = 1 \rightarrow w \cdot x + b - 1 = 0$ con vector normal w y distancia perpendicular al origen:

$$\frac{|1 - b|}{\|w\|} \tag{2.7}$$

donde $\|w\|$ es la norma euclídea de w, para el otro caso, es decir que pertenezcan a la clase 2, caen en el hiperplano $H_2 : w \cdot x + b = -1 \rightarrow w \cdot x + b + 1 = 0$, entonces la distancia perpendicular al origen en este caso sería:

$$\frac{|-1 - b|}{\|w\|} \tag{2.8}$$

Ahora la distancia del hiperplano H_1 al otro hiperplano H_2 es la suma de las dos expresiones anteriores, es decir:

$$D = \frac{|1 - b|}{\|w\|} + \frac{|-1 - b|}{\|w\|} = \begin{cases} \frac{(1-b)+(1+b)}{\|w\|} & \text{si } -1 < b < 0 \\ \frac{(1-b)+(1+b)}{\|w\|} & \text{si } 0 < b < 1 \end{cases} = \frac{2}{\|w\|} \tag{2.9}$$

Como H_1 y H_2 son paralelos y no hay datos de entrenamiento entre ellos, la separación entre éstos es la suma de las distancias de los resultados de (2.7) y (2.8), a esto se le conoce como margen, es decir:

$$M = \frac{2}{\|w\|} \tag{2.10}$$

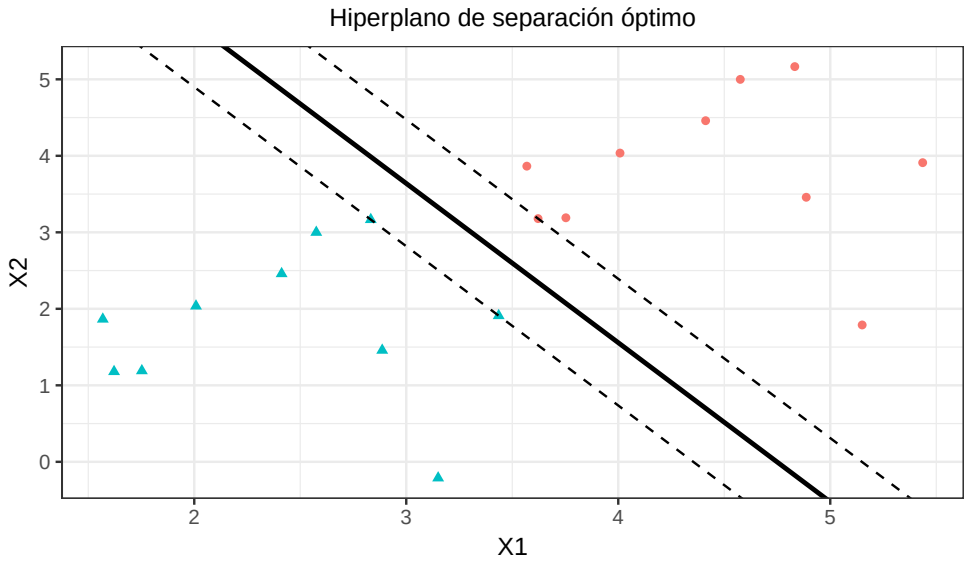


Figura 2.2: Márgen máximo (líneas punteadas) e hiperplano de separación óptimoen un conjunto linealmente separable (línea remarcada).

Este resultado lo podemos ver en la Figura 2.2 en donde observamos que la distancia del márgen máximo al hiperplano de separación óptimo en un conjunto linealmente separable es igual a $\frac{1}{\|w\|}$.

Este resultado será indispensable para poder idear el algoritmo de aprendizaje que maximice el margen, nos llevará a resolver un problema de programación cuadrática, que veremos en breve. De forma más general, con el resultado anterior todos los datos de entrenamiento deben satisfacer:

$$\frac{y_j D(x_j)}{\|w\|} \geq M \text{ para } j = 1, ..., n \tag{2.11}$$

donde $D(x_j)$ es la función de decisión para el dato de entrenamiento j. Para poder hallar el hiperplano de separación óptimo lo que debemos hacer es encontrar el valor de w que haga que maximice el margen, sin embargo,

existen infinidad de soluciones que solo difieren en la escala de w, para poder limitar el número de soluciones a una sola y teniendo en cuenta que (2.11) se puede expresar también como:

$$y_j D(x_j) \geq M\|w\| \quad \text{para} \quad j = 1, ..., n \tag{2.12}$$

Si la escala de $M\|w\|$ es igual a la unidad, es decir:

$$M\|w\| = 1 \tag{2.13}$$

entonces con esta restricción lo que queremos es que se minimice la norma euclidiana que satisface (2.13) y se puede hallar resolviendo el problema para minimizar w y b:

$$\underset{w,b}{\text{minimizar}} \quad \frac{1}{2}\|w\|^2 \tag{2.14}$$

$$\text{sujeto a} \quad y_i(w^T x_i + b) \geq 1, \ i = 1, \ldots, n. \tag{2.15}$$

El problema anterior es un problema de programación cuadrática convexa que cumple con lo siguiente:

- Si decimos que hay separabilidad lineal entonces existen w y b que satisfacen (2.14).
- Llamamos soluciones factibles a las que satisfacen (2.15).
- El valor de la función objetivo es único, es decir, la unicidad para las máquinas de soporte vectorial no es ningún problema y es una ventaja que poseen.
- El número de variables que tiene este problema es el número de las variables de entrada más uno (i.e. $n + 1$), cabe mencionar que cuando el número de variables es pequeño entonces podemos resolverlo como un problema de programación cuadrática, pero como el espacio de entrada puede llegar a tener una alta dimensión se debe convertir al problema dual en donde el número de variables será el número de datos de entrenamiento.

Para poder realizar esto debemos hallar la solución al problema de optimización de la ecuación (2.14) bajo las restricciones de (2.15) de la siguiente forma: minimizando w, maximizando los α_i y minimizando o maximizando

a b, es decir, encontrar el punto silla de la función Lagrangiana (ecuación (2.16))

$$L(w,b,\alpha) = \frac{1}{2}w^T w - \sum_{i=1}^{n} \alpha_i\{y_i(w^T x_i + b) - 1\} \tag{2.16}$$

donde $\alpha = (\alpha_1, ..., \alpha_n)^T$ y los $\alpha_i \geq 0$ son los multiplicadores de Lagrange. Ahora debemos aplicar las condiciones de Karush-Kuhn-Tucker, también conocidas como condiciones KKT:

$$\frac{\partial L(w,b,\alpha)}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0,\ i = 1, ..., n \tag{2.17}$$

$$\frac{\partial L(w,b,\alpha)}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0,\ i = 1, ..., n \tag{2.18}$$

$$\alpha_i\{y_i(w^T x_i + b) - 1\} = 0,\ i = 1, ..., n \tag{2.19}$$

La dualidad Lagrangiana nos permite transformar el problema primal, ecuación (2.16), en su problema dual, que es más sencillo de resolver. Las ecuaciones (2.17) y (2.18), nos dan respectivamente los siguientes resultados:

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i,\ i = 1, ..., n \tag{2.20}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.21}$$

Las dos expresiones anteriores las sustituimos en la ecuación (2.16) y nos queda el siguiente problema de optimización dual:

$$\text{maximizar } W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{2.22}$$

$$\text{sujeto a} \quad \sum_{i=1}^{n} \alpha_i y_i = 0,\ \alpha_i \geq 0 \quad \text{para} \quad i = 1, \ldots, n \tag{2.23}$$

La solución α^* del problema dual nos permitirá conocer la solución del problema primal, es decir, sólo habrá que sustituir dicha solución en (2.20) y, finalmente, sustituir el resultado obtenido en (2.3), tendríamos entonces:

$$D(x) = \sum_{i \epsilon S} \alpha_i y_i x_i^T x + b \tag{2.24}$$

donde S es el conjunto de los vectores de soporte. En la solución, a cada dato de entrenamiento le corresponde un multiplicador de Lagrange óptimo α_i^*, los

datos para los cuales $\alpha_i^* > 0$ son los vectores de soporte y están sobre alguno de los hiperplanos H_1 y H_2. Los demás datos de entrenamiento cumplen con $\alpha_i = 0$ y son los que están a los lados de H_1 y H_2.

Para las MSV, los vectores de soporte son los elementos más importantes de los datos de entrenamiento, ya que el hiperplano de separación (2.24), será una combinación lineal de sólo los vectores de soporte del conjunto total de datos de entrenamiento.

Los vectores de soporte cumplen como igualdad a la condición (2.3) a cada uno le podemos calcular un valor para el párametro b, despejando a b de (2.3), obtenemos lo siguiente:

$$b = y_i - w^T x_i \quad \text{para} \quad i \epsilon S \tag{2.25}$$

Para poder obtener un cálculo más preciso, es mejor calcular el promedio al conjunto de vectores de soporte, es decir:

$$b = \frac{1}{|S|} \sum_{i \epsilon S} (y_i - w^T x_i) \tag{2.26}$$

En MSV, los valores de los multiplicadores de Lagrange tienen influencia sobre los vectores de soporte, es decir, entre mayor sea α_i^* mayor será su influencia sobre el hiperplano de separación.

A continuación se mostrarán un par de ejemplos en donde se podrá comprobar lo dicho anteriormente.

2.1.2. Ejemplo 1: conjunto linealmente separable en una dimensión

Consideremos un caso linealmente separable en una sola dimensión, como se muestra en la figura 2.3. Las restricciones de desigualdad de acuerdo a (2.15) son las siguientes:

$$-w + b \geq 1 \tag{2.27}$$

$$-b \geq 1 \tag{2.28}$$

$$-(w + b) \geq 1 \tag{2.29}$$

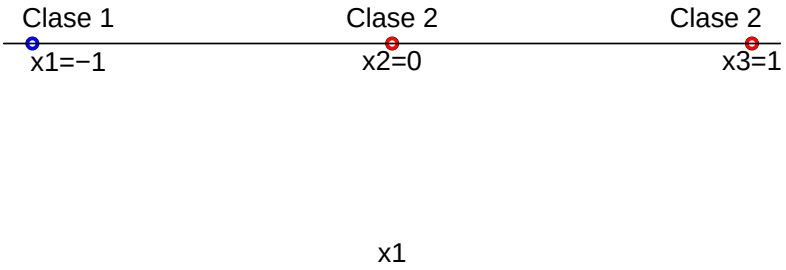


Figura 2.3: Ejemplo 1: datos linealmente no separables en una dimensión.

La región de (w, b) que satisface a las restricciones (2.27), (2.28) y (2.29) es el área que está marcada con los rombos azules de la figura 2.4, entonces las soluciones que minimizan a $\|w\|^2$ sujetas a las restricciones de (2.15) son:

$$b = -1, \quad w = -2$$

Por lo tanto la función de decisión sería la siguiente:

$$D(x) = -2x - 1 \tag{2.30}$$

Por consiguiente el punto que separa correctamente a las dos clases es $x = -\frac{1}{2}$. Como la solución está dada por (2.27) y (2.28), $x = 0$ y $x = -1$ son los vectores de soporte.

El problema dual está dado de la siguiente manera:

$$\begin{aligned} \text{maximizar} \quad & W(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(\alpha_1 + \alpha_3)^2 && (2.31)\\ \text{sujeto a} \quad & \alpha_1 - \alpha_2 - \alpha_3 = 0 && (2.32)\\ & \alpha_i \geq 0 \quad \text{para} \quad i = 1, 2, 3. && (2.33) \end{aligned}$$

De (2.32) tenemos que $\alpha_2 = \alpha_1 - \alpha_3$, sustituyendo en (2.31) nos queda lo siguiente:

$$\begin{aligned} & W(\alpha) = 2\alpha_1 - \tfrac{1}{2}(\alpha_1 + \alpha_3)^2 && (2.34)\\ & \alpha_i \geq 0 \quad \text{para} \quad i = 1, 2, 3. && (2.35) \end{aligned}$$

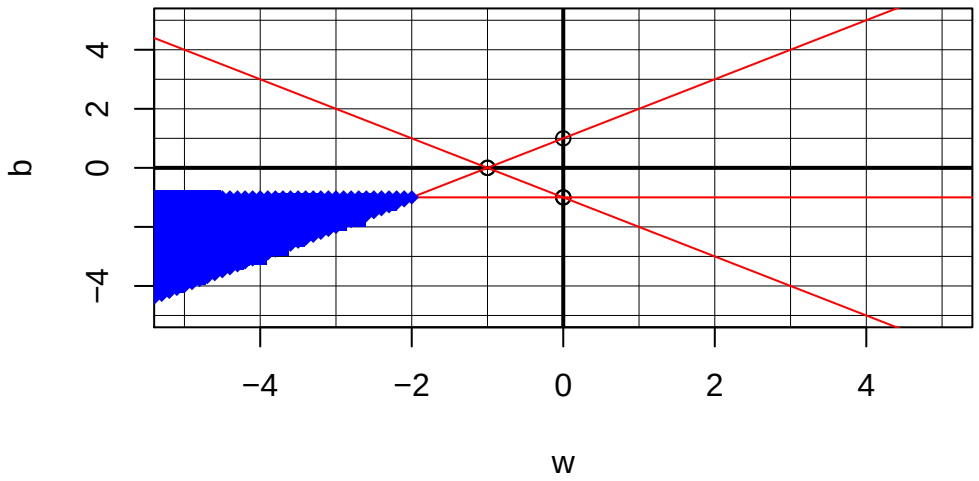


Figura 2.4: Ejemplo 1: región factible que cumple con las restricciones (2.27), (2.28) y (2.29)

Como α_1, α_3 son ≥ 0, la función (2.34) es maximizada cuando $\alpha_3 = 0$. Por lo tanto (2.34) se reduce a:

$$W(\alpha) = 2\alpha_1 - \tfrac{1}{2}(\alpha_1)^2 = -\tfrac{1}{2}(\alpha_1 - 2)^2 + 2 \tag{2.36}$$

$$\alpha_1 \geq 0 \tag{2.37}$$

Debido a que la función (2.36) se maximiza para $\alpha_1 = 2$, la solución óptima para (2.31) es:

$$\alpha_1 = 2,\ \alpha_2 = 2,\ \alpha_3 = 0$$

Por lo tanto $x = -1$ y $x = 0$ son vectores de soporte y $w = -2, \quad b = -1$, como vemos obtuvimos el mismo resultado que al resolver el problema primal.

2.1.3. Ejemplo 2: conjunto linealmente separable en dos dimensiones

Supongamos que tenemos el siguiente conjunto de datos presentados en la tabla 2.1. La figura 2.5 grafica estos datos, donde los círculos azules son datos positivos y los círculos rojos son datos negativos.

x_1	x_2	y_i
3	1	+1
3	-1	+1
6	1	+1
6	-1	+1
1	0	-1
0	1	-1
0	-1	-1
-1	0	-1

Tabla 2.1: Ejemplo 2: datos.

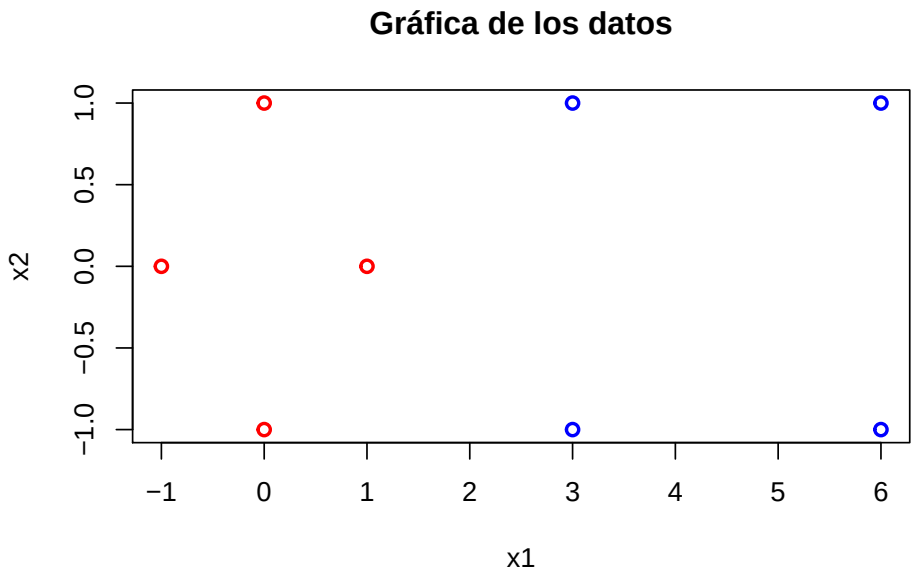


Figura 2.5: Ejemplo 2: datos linealmente separables en dos dimensiones.

Nos gustaría poder encontrar una MSV que separe con precisión a las dos clases, ya que los datos son linealmente separables, podemos usar una MSV lineal (es decir, una cuya función es la función de identidad). Si observamos en la figura 2.5, los puntos más adecuados para poder hallar un hiperplano de separación son los puntos $(1, 0), (3, -1)$ y $(3, 1)$, y serían los vectores de soporte. La figura 2.6 muestra los vectores de soporte señalados con cuadros amarillos.

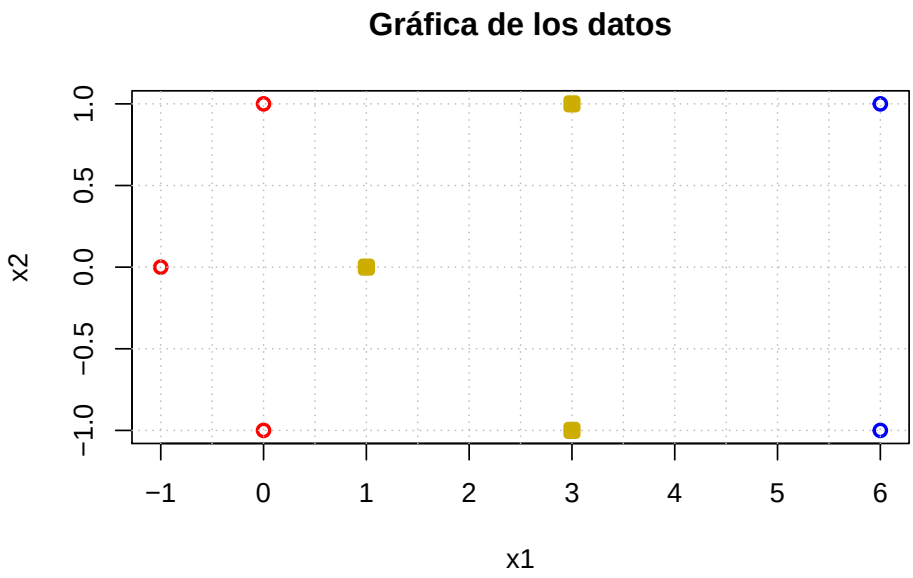


Figura 2.6: Ejemplo 2: vectores de soporte señalados con cuadros amarillos

A continuación utilizaremos vectores aumentados con un sesgo igual a 1. Es decir:

- Si $s_1 = (1, 0)$, entonces $\tilde{s_1} = (1, 0, 1)$
- Si $s_2 = (3, 1)$, entonces $\tilde{s_2} = (3, 1, 1)$
- Si $s_3 = (3, -1)$, entonces $\tilde{s_3} = (3, -1, 1)$

Nuestra tarea es encontrar valores para el α_i tal que:

$$\begin{aligned}
\alpha_1\phi(s_1)\cdot\phi(s_1) + \alpha_2\phi(s_2)\cdot\phi(s_1) + \alpha_3\phi(s_3)\cdot\phi(s_1) &= -1 \\
\alpha_1\phi(s_1)\cdot\phi(s_2) + \alpha_2\phi(s_2)\cdot\phi(s_2) + \alpha_3\phi(s_3)\cdot\phi(s_2) &= +1 \\
\alpha_1\phi(s_1)\cdot\phi(s_3) + \alpha_2\phi(s_2)\cdot\phi(s_3) + \alpha_3\phi(s_3)\cdot\phi(s_3) &= +1
\end{aligned}$$

Puesto que $\phi(\cdot)$ denota al vector aumentado, es decir $\phi(s_1) = \tilde{s_1}$, lo anterior se resume en:

$$\begin{aligned}
\alpha_1\tilde{s_1}\cdot\tilde{s_1} + \alpha_2\tilde{s_2}\cdot\tilde{s_1} + \alpha_3\tilde{s_3}\cdot\tilde{s_1} &= -1 \\
\alpha_1\tilde{s_1}\cdot\tilde{s_2} + \alpha_2\tilde{s_2}\cdot\tilde{s_2} + \alpha_3\tilde{s_3}\cdot\tilde{s_2} &= +1 \\
\alpha_1\tilde{s_1}\cdot\tilde{s_3} + \alpha_2\tilde{s_2}\cdot\tilde{s_3} + \alpha_3\tilde{s_3}\cdot\tilde{s_3} &= +1
\end{aligned}$$

Resolviendo las ecuaciones anteriores para α_i nos queda lo siguiente:

$$\begin{aligned} 2\alpha_1 + 4\alpha_2 + 4\alpha_3 &= -1 \\ 4\alpha_1 + 11\alpha_2 + 9\alpha_3 &= +1 \\ 4\alpha_1 + 9\alpha_2 + 11\alpha_3 &= +1 \end{aligned}$$

Resolviendo el sistema de ecuaciones, nos quedan las soluciones siguientes:

$$\alpha_1 = -3.5\ ,\ \alpha_2 = 0.75 \quad \text{y} \quad \alpha_3 = 0.75$$

Ahora que tenemos los α_i, ¿cómo localizamos el hiperplano que discrimina el positivo de los ejemplos negativos?. Para esto tenemos que:

$$w = \sum_i \alpha_i \tilde{s}_i, \quad \text{con} \quad i = 1, 2, 3 \tag{2.38}$$

$$-3.5 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix}$$

Por lo tanto, como anteriormente aumentamos a los vectores con un sesgo de 1, podemos equiparar la última entrada en w como el sesgo del hiperplano óptimo, es decir a b, y así escribir la ecuación del hiperplano de separación óptimo $y = w \cdot x + b$ con $w = \binom{1}{0}$ y $b = -2$. Como el hiperplano de separación óptimo cumple con la condición (2.15). Entonces la ecuación nos quedaría de la siguiente manera:

$$(1, 0)x - 2 = 0 \tag{2.39}$$

Por consiguiente el hiperplano de separación óptimo se muestra en la figura 2.7. Lo cual implica que el hiperplano de separación óptimo de acuerdo a la solución de la ecuación 2.39 es aquel tal que $x_1 = 2$.

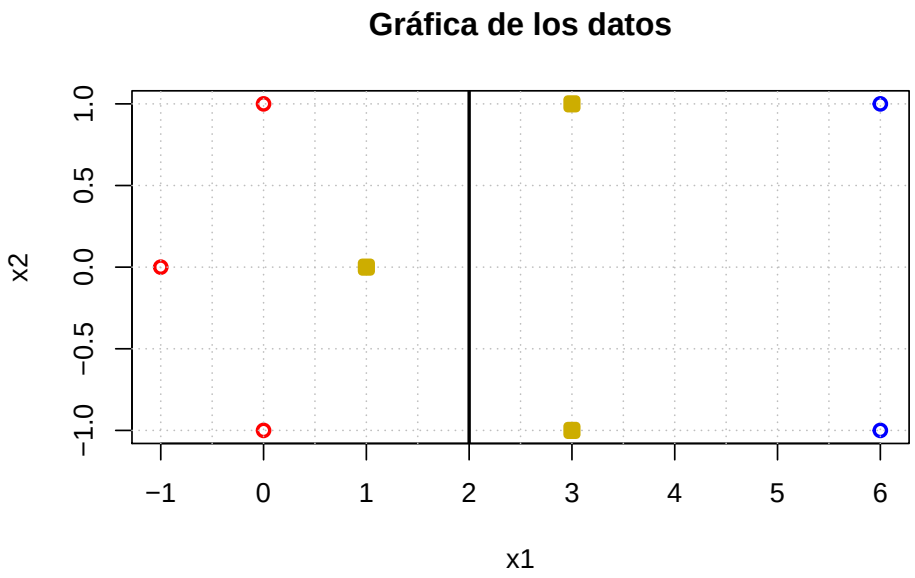


Figura 2.7: Ejemplo 2: hiperplano de separación óptimo.

2.1.4. Clasificador lineal de margen suave

En la realidad no es común tener ejemplos con datos separables, es decir, se encuentran vectores que pertenecen a una clase, dentro de la región de los vectores que pertenecen a otra clase y por consiguiente nunca podrían ser separados por medio de algún hiperplano. Por lo tanto, el problema de optimización (2.14) no podría encontrar una solución factible y tampoco podría satisfacer la desigualdad (2.15). Para cualquier punto de datos de entrenamiento mal clasificado x_i, el α_i correspondiente tiende a infinito, este punto en particular ejerce una influencia fuerte en el límite de decisión para ser clasificado correctamente. Cuando el valor de α_i llega a su máximo ya no puede aumentar su influencia y entonces es mal clasificado, sin embargo, cuando sucede esto el algoritmo elige casi todos los puntos de los datos de entrenamiento como vectores de soporte. En la práctica se permite un margen suave (ya sea en el lado correcto de la línea de separación o en el equivocado). Sin embargo, no es difícil pasar del caso separable al caso no separable introduciendo un nuevo conjunto de variables de holgura $\{\xi_i\}_{i=1}^{n}$ en las restricciones, las cuales miden la cantidad de error por no cumplir con las restricciones. Estas variables son no negativas ($\xi_i \geq 0$) y se agregan en la restricción (2.15),

es decir:

$$y_i(w^T x_i + b) \geq 1 - \xi_i,\ i = 1, \ldots, n. \tag{2.40}$$

Para poder obtener el hiperplano óptimo debemos:

$$\text{minimizar} \quad \sum_{i=1}^{n} \theta(\xi_i) \tag{2.41}$$

$$\text{donde} \qquad \theta(\xi_i) = \begin{cases} 1 & \text{si } \xi_i \geq 0 \\ 0 & \text{si } \xi_i = 0 \end{cases} \tag{2.42}$$

Sin embargo, el problema anterior es complicado de resolver, lo que debemos considerar es lo siguiente:

$$\underset{w,b,\xi}{\text{minimizar}} \quad \frac{1}{2}\|w\|^2 + \frac{C}{p}\sum_{i=1}^{n}(\xi_i)^p \tag{2.43}$$

$$\text{sujeto a} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n. \tag{2.44}$$

Donde w es el vector n-dimensional, b es el sesgo, $\xi = (\xi_1, \ldots, \xi_n)^T$ es la variable de holgura para x_i, $p = 1, 2$ y C es el parámetro del margen y es una constante suficientemente grande elegida por el usuario, que permite regular el compromiso entre la dimensión VC y la proporción del número de datos no separables. Para poder interpretar el valor de C, tenemos lo siguiente:

- Cuando $C \to \infty$ y $\xi \to 0$, los datos serían perfectamente separables.
- Cuando C tiene un valor pequeño entonces los ξ_i tendrían un valor muy grande, es decir, se estarían admitiendo un número muy alto de datos mal clasificados.
- Si $C \to 0$ y $\xi \to \infty$ todos los datos estarían mal clasificados.

El hiperplano de separación se denomina hiperplano de separación de margen suave, cuando $p = 1$ a la MSV se le denomina L1 y cuando $p = 2$ a la MSV se le denomina L2 que más adelante estudiaremos. Para falicitar la resolución igual que en el problema anterior obtenemos la función Lagrangiana cuando $p = 1$, es decir, MSV L1 (ecuación 2.45), para así poder calcular posteriormente el problema dual.

$$L(w, b, \alpha, \xi, \beta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\{y_i(w^T(x_i) + b) - 1 + \xi_i\} - \sum_{i=1}^{n}\beta_i\xi_i \tag{2.45}$$

donde $\alpha = (\alpha_1, \ldots, \alpha_n)^T$ y $\beta = (\beta_1, \ldots, \beta_n)^T$ deben satisfacer la condición de ser no negativas, al igual que el parámetro ξ_i.

Para la solución óptima se cumplen las siguientes condiciones KKT:

$$\frac{\partial L(w, b, \alpha, \xi)}{\partial w} = 0 \tag{2.46}$$

$$\frac{\partial L(w, b, \alpha, \xi)}{\partial \xi_i} = 0 \tag{2.47}$$

$$\frac{\partial L(w, b, \alpha, \xi)}{\partial b} = 0 \tag{2.48}$$

$$\alpha_i\{y_i(w^T x_i + b) - 1 + \xi_i\} = 0,\ i = 1, \ldots, n \tag{2.49}$$

$$\beta_i \xi_i = 0,\ i = 1, \ldots, n \tag{2.50}$$

$$\alpha_i, \beta_i, \xi_i \geq 0,\ i = 1, \ldots, n \tag{2.51}$$

Usando (2.45), se reduce de (2.46) a (2.48), respectivamente a:

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad i = 1, \ldots, n \tag{2.52}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.53}$$

$$\alpha_i + \beta_i = C, \quad i = 1, \ldots, n \tag{2.54}$$

Sustituyendo (2.52), (2.53) y (2.54) en (2.45), obtenemos el problema dual:

$$\text{maximizar } W(\alpha) \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{2.55}$$

$$\text{sujeto a} \qquad \sum_{i=1}^{n} \alpha_i y_i = 0,\ C \geq \alpha_i \geq 0 \text{ para } \ i = 1, \ldots, n \tag{2.56}$$

La diferencia entre las MSV de margen suave L1 y las MSV de margen máximo es que en este caso α_i no puede exceder de C, la desigualdad en (2.56) se denomina restricción de caja.

De (2.50) se derivan tres casos para α_i:

1. Si $\alpha_i = 0 \rightarrow \xi_i = 0$ y se clasifica correctamente.
2. Si $0 < \alpha_i < C \rightarrow \{y_i(w^T x_i + b) - 1 + \xi_i\} = 0, \quad \xi_i = 0$. Por lo tanto x_i es un vector de soporte fuera de la frontera.

3. Si $\alpha_i = C \rightarrow \{y_i(w^T x_i + b) - 1 + \xi_i\} = 0, \quad \xi_i \geq 0$, entonces x_i es un vector de soporte dentro de la frontera. Si $0 \leq \xi_i < 1$, x_i es correctamente clasificado y si $\xi_i \geq 1$, x_i está mal clasificado.

La función de decisión es la misma que la de margen máximo, la cual está dada por:

$$D(x) = \sum_{i \epsilon S} \alpha_i y_i x_i^T x + b \tag{2.57}$$

donde S es el conjunto de los vectores de soporte, ya que $\alpha_i \neq 0$ para los vectores de soporte, la suma en (2.57) es sólo para los vectores de soporte. Para el α ilimitado tenemos que:

$$b = y_i - w^T x_i \quad \text{para} \quad i \epsilon S \tag{2.58}$$

Para proteger la seguridad de los cálculos hacemos el promedio de b, el cual se calcula para vectores de soporte ilimitados:

$$b = \frac{1}{|U|} \sum_{i \epsilon U} (y_i - w^T x_i) \tag{2.59}$$

donde U es el conjunto de índices vectoriales de soporte ilimitados.

Para L2 en el segundo término de la función de coste (2.43) se usa $p = 2$, es decir:

$$\underset{w,b,\xi}{\text{minimizar}} \quad \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} (\xi_i)^2 \tag{2.60}$$

$$\text{sujeto a} \quad y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n. \tag{2.61}$$

Donde w es el vector l-dimensional, b es el sesgo, $\phi(x)$ es la función de mapeo que mapea el vector n-dimensional x en el espacio característico l-dimensional, ξ_i es la variable de holgura para x_i y C es el parámetro del margen, obtenemos la función Lagrangiana, entonces:

$$L(w, b, \alpha, \xi) = \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 - \sum_{i=1}^{n} \alpha_i \{y_i(w^T \phi(x_i) + b) - 1 + \xi_i\} \tag{2.62}$$

En esta función no necesitamos agregar los multiplicadores de Lagrange para ξ_i, debido a la condición (2.64), tenemos que $C = \alpha_i$. Como vemos para L2 es más importante minimizar el error, es decir, L2 requiere que los datos se clasifiquen mejor y haya menos penalización para ello. Aquí ξ_i y α_i son

no negativos. Para obtener la solución óptima usamos las condiciones KKT como sigue:

$$\frac{\partial L(w,b,\alpha,\xi)}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i \phi(x_i) = 0 \tag{2.63}$$

$$\frac{\partial L(w,b,\alpha,\xi)}{\partial \xi_i} = C - \alpha_i = 0 \tag{2.64}$$

$$\frac{\partial L(w,b,\alpha,\xi)}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.65}$$

$$\alpha_i\{y_i(w^T\phi(x_i)+b) - 1 + \xi_i\} = 0,\ i = 1,\ldots,n \tag{2.66}$$

Los puntos x_i con $\alpha_i \geq 0$ son los vectores de soporte. Anteriormente los valores absolutos de la función de decisión evaluada en los vectores de soporte son 1, sin embargo, para este caso, debido a la condición (2.64) los márgenes de los vectores de soporte son menores que 1. La condición (2.66) da las condiciones de complementariedad KKT y de (2.63), (2.64) y (2.66), la solución óptima debe satisfacer $\alpha_i = 0$ ó:

$$y_i \left(\sum_{j=1}^{n} \alpha_j y_j \left(k(x_j, x_i) + \frac{\delta_{ij}}{C} \right) + b \right) - 1 = 0 \tag{2.67}$$

Donde $k(x, x') = \phi^T(x)\phi(x)$ y δ_{ij} es la función delta de Kronecker, la cual está definida de la siguiente manera:

$$\delta_{ij} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases} \tag{2.68}$$

Con esto podemos calcular el término b para $\alpha_i \geq 0$:

$$b = y_j - \sum_{i \epsilon S} \alpha_i y_i \left(k(x_i, x_j) + \frac{\delta_{ij}}{C} \right), \tag{2.69}$$

Donde x_j es un vector de soporte y S es el conjunto de vectores de soporte, para dar un mejor resultado en los cálculos hacemos el promedio de la siguiente forma:

$$b = \frac{1}{|S|} \left(y_j - \sum_{i \epsilon S} \alpha_i y_i \left(k(x_i, x_j) + \frac{\delta_{ij}}{C} \right) \right), \tag{2.70}$$

Las funciones (2.69) y (2.70) son diferentes a las de L1, debido a que se agregó la función (2.68), sin embargo la función de decisión sigue siendo la misma:

$$D(x) = \sum_{i \epsilon S} \alpha_i y_i x_i^T x + b \tag{2.71}$$

Para obtener el problema dual debemos sustituir las ecuaciones (2.63), (2.64) y (2.65) en la función Lagrangiana (2.62):

$$\text{maximizar } W(\alpha) \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left(k(x_i, x_j) + \frac{\delta_{ij}}{C} \right) \tag{2.72}$$

$$\text{sujeto a} \qquad \sum_{i=1}^{n} \alpha_i y_i = 0,\ \alpha_i \geq 0 \text{ para } i = 1, \ldots, n \tag{2.73}$$

Como se agrega la función $\frac{\delta_{ii}}{C}$, el problema de optimización asociado es más estable desde el punto de vista computacional que el de la máquina de vector de soporte L1, debido a esto las MSV de margen suave L2 son muy parecidas a las MSV de margen máximo, es por ello que podemos hacer lo siguiente:

$$\tilde{w} = \begin{pmatrix} w \\ \sqrt{\frac{C}{2}}\xi \end{pmatrix},\ \tilde{b} = b, \quad \tilde{\phi}(x_i) = \begin{pmatrix} \phi(x_i) \\ \sqrt{\frac{2}{C}} y_i e_i, \end{pmatrix}$$

aquí e_i es el vector n-dimensional donde el i-ésimo elemento es 1 y todos los demás son 0, por lo tanto el problema de optimización para MSV L2 queda de la siguiente forma:

$$\text{minimizar} \quad \frac{1}{2} \tilde{w}^T \tilde{w} \tag{2.74}$$

$$\text{sujeto a} \quad y_i(\tilde{w}^T \tilde{\phi}(x_i) + \tilde{b}) \geq 1,\ i = 1, \ldots, n. \tag{2.75}$$

Entonces la MSV L2 es equivalente a la MSV de margen máximo con el espacio de características aumentado, ya que siempre tendrá una solución debido a las variables de holgura, esto significa que la solución no se superpone en el espacio de características aumentadas.

2.1.5. Caso no lineal

Los dos casos que vimos anteriormente son muy limitados, ya que en realidad aunque se pueda encontrar una solución factible aún así se pueden superponer las clases y mostrar error en la clasificación de los datos, hasta ahora se ha asumido que los datos son separables, en esta sección se estudiará cómo usar conjuntos de funciones no lineales, para definir espacios transformados de alta dimensionalidad y cómo buscar hiperplanos de separación óptimos en

dichos espacios transformados, a estos espacios se les denomina espacio de características, que definiremos más adelante. Si usamos una separabilidad no lineal entonces estaríamos clasificando sin ningún error a las dos clases.

La idea de una MSV no lineal es mapear los vectores de entrada $x\epsilon R^n$ en vectores $\phi(x)\epsilon R^l$ de un espacio de características con una dimensión más grande, donde ϕ representa el mapeo de $R^n \rightarrow R^l$, entonces lo que debemos hacer es resolver el siguiente problema de clasificación lineal en este espacio de funciones:

$$x\epsilon R^n \rightarrow \phi(x) = \{\phi_1(x), \phi_2(x), \ldots, \phi_n(x)\}^T \epsilon R^l \tag{2.76}$$

Entonces la función de decisión lineal en el espacio de características nos quedaría de la siguiente forma:

$$D(x) = w^T \phi(x) + b \tag{2.77}$$

donde w es el vector l-dimensional y b es el sesgo. En su forma dual, la función de decisión se obtiene transformando convenientemente la expresión (2.77) en:

$$D(x) = \sum_{i\epsilon S} \alpha_i y_i K(x, x_i) + b \tag{2.78}$$

donde $K(x, x')$ es una función núcleo, la cual asigna a cada par de elementos del espacio de características un valor real, más detalles se describen en la sección 2.2. La ventaja de usar núcleos es que no se necesita tratar detalladamente el espacio de características que tengan una dimensión alta, a esto se le llama truco del núcleo. Usando el núcleo, el problema dual en el espacio de características se da como sigue:

$$\text{maximizar } W(\alpha) \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{2.79}$$

$$\text{sujeto a} \qquad \sum_{i=1}^{n} \alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C \text{ para } \ i = 1, \ldots, n \tag{2.80}$$

La figura 2.8 muestra cómo la función ϕ mueve los datos de entrada x_i (no linealmente separables) a un espacio de mayor dimensión para poder separarlos linealmente.

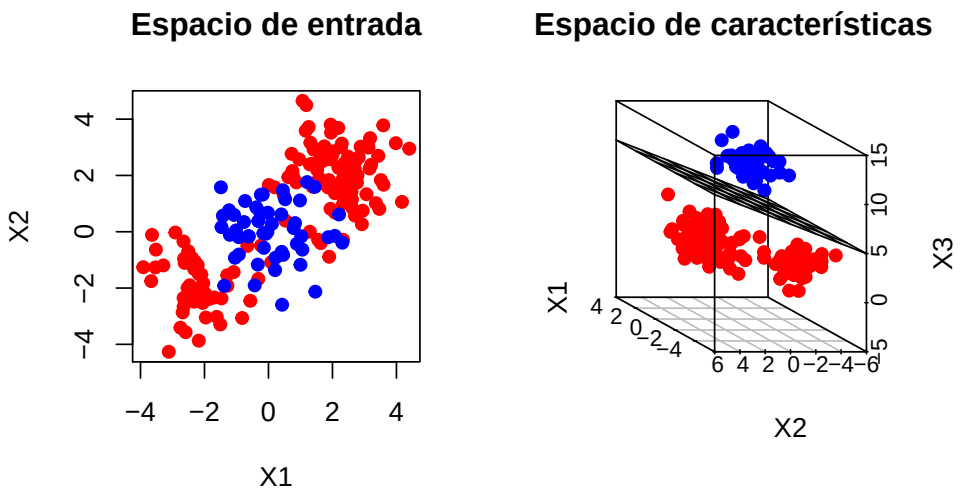


Figura 2.8: Datos no separables linealmente. Espacio de entrada y espacio de características de datos no separables linealmente.

Como $K(x, x')$ es un núcleo positivo semidefinido (para más detalle ver la sección 2.2), el problema de optimización es un problema de programación cuadrática cóncava y como $\alpha_i = 0$ entonces hay solución factible. Las condiciones de complementariedad KKT están dadas por:

$$\alpha_i \left(y_i \left(\sum_{j=1}^{n} \alpha_j y_j k(x_i, x_j) + b \right) - 1 + \xi_i \right) = 0 \tag{2.81}$$

$$(C - \alpha_i)\xi_i = 0 \tag{2.82}$$

$$\alpha_i, \xi_i \geq 0 \tag{2.83}$$

Con (2.77) podemos calcular el término b como sigue:

$$b = y_j - \sum_{i \epsilon S} \alpha_i y_i k(x_i, x_j), \tag{2.84}$$

donde x_j es un vector de soporte, para dar un mejor resultado en los cálculos hacemos el promedio de la siguiente forma:

$$b = \frac{1}{|U|} \sum_{j \epsilon U} \left(y_j - \sum_{i \epsilon S} \alpha_i y_i k(x_i, x_j) \right), \tag{2.85}$$

donde U es el conjunto de vectores de soporte. Anteriormente habíamos mencionado otros conjuntos de vectores de soporte denominados con las letras B

y S, éstos están definidos de la siguiente manera:

$$U = \{i|0 < \alpha_i < C\}, \quad \text{para} \quad i = 1, \ldots, n \tag{2.86}$$
$$B = \{i|\alpha_i = C\}, \quad \text{para} \quad i = 1, \ldots, n \tag{2.87}$$
$$S = U \cup B \tag{2.88}$$

La regla de decisión de los tres casos que mencionamos anteriormente es la siguiente:

$$x \in \begin{cases} Clase1 & \text{si } D(x) > 0 \\ Clase2 & \text{si } D(x) < 0 \\ No \quad clasificable & \text{si } D(x) = 0 \end{cases} \tag{2.89}$$

2.2. Función Núcleo

Una función núcleo (*kernel*) es de la forma $K : X \times X \to R$ que asigna a cada par de elementos del espacio de entrada, X, un valor real correspondiente al producto escalar de dichos elementos en un nuevo espacio L (espacio de características), es decir:

$$k(x, x') = \phi^T(x)\phi(x') \tag{2.90}$$

las ventajas de poder usar núcleos en una máquina de soporte vectorial son las siguientes:

- Se puede mejorar el rendimiento de la generalización en el algoritmo de clasificación.
- La complejidad del problema de optimización sigue dependiendo solamente de la dimensionalidad del espacio de entrada y no del espacio de características, con esto es posible operar en un espacio de características con una dimensión alta.
- Los productos escalares requeridos en un espacio de características ($\phi^T(x_i)\phi(x_j)$), se calculan directamente utilizando los núcleos $k(x_i, x_j)$ para vectores de datos de entrenamiento dados en un espacio de entrada, sin embargo, cabe mencionar que el cálculo de este producto escalar puede ser desalentador desde el punto de vista computacional si el número de características l (es decir, la dimensionalidad l de un espacio de características) es muy grande.

Dado que cada núcleo tiene cierto grado de variabilidad en la práctica se debe experimentar con diferentes núcleos y ajustar sus parámetros a través de la búsqueda de modelos para minimizar el error en un conjunto de pruebas.

La función núcleo necesita satisfacer las condiciones de Mercer. De acuerdo con la teoría de Hilbert-Schmidt, si la función asimétrica $k(x, x')$ satisface

$$\sum_{i,j=1}^{n} h_i h_j k(x_i, x_j) = \left(\sum_{i=1}^{n} h_i \phi^T(x_i) \right) \left(\sum_{j=1}^{n} h_j \phi(x_j) \right) \geq 0 \qquad (2.91)$$

donde n es un número natural y h_i toma valores reales, la función que satisface (2.91) se denomina núcleo positivo semidefinido o núcleo de Mercer. En lo que sigue, si no hay confusión, simplemente lo llamaremos núcleo. Esto implica que la matriz del núcleo de n por n, en la que la entrada (i, j) es $K(x_i, x_j)$, es siempre positiva semidefinida.

Con base en lo anterior, para verificar que cierta función es una función núcleo el procedimiento es el siguiente:

- Construir de forma explícita un espacio de características L (el cual debe ser una espacio de Hilbert, el cual permite que nociones y técnicas algebraicas y geométricas aplicables a espacios de dimensión dos y tres se extiendan a espacios de dimensión arbitraria, incluyendo a espacios de dimensión infinita.).
- Establecer también explícitamente una función $\phi : X \to L$.
- Comprobar que se cumple que $K(x_i, x_j) = (\phi^T(x_i)\phi(x_j))$.

Por ejemplo si aplicamos este procedimiento a $x \in R^2$, es decir, $x = [x_1, x_2]^T$ y denotamos a $\phi(x) = [x_1^2, \sqrt{2}x_1x_2, x_1^2]^T$, el producto punto estaría definido de la siguiente manera:

$$\begin{aligned} (\phi^T(x_i)\phi(x_j)) &= [x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2], [x_{j1}^2, \sqrt{2}x_{j1}x_{j2}, x_{j2}^2]^T \\ &= [x_{i1}^2 x_{j1}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} + x_{i2}^2 x_{j2}^2] \\ &= (x_i^T x_j)^2 = k(x_i, x_j) \end{aligned}$$

por lo que el espacio de características L sería R^3, $\phi : R^2 \to R^3 : \phi(x) = [x_1^2, \ \sqrt{2}x_1x_2, \ x_1^2]^T$ y $k(x_i, x_j) = (\phi^T(x_i)\phi(x_j))$.

Como mencionamos anteriormente existen distintos tipos de núcleos con los que se pueden clasificar correctamente los datos no lineales, sin embargo sólo mostraremos los de mayor importancia y los que son más usados.

Núcleo Lineal

Si tenemos un problema de clasificación que se puede separar linealmente en el espacio de entrada, no es necesario utilizar la función de mapeo, en este caso solo utilizamos núcleos lineales, es decir:

$$k(x, x') = x^T x' \tag{2.92}$$

Núcleo Polinomial

El mapeo polinomial es un método popular para separar datos que no son lineales, el núcleo polinomial con grado d está dado por:

$$k(x, x') = (x^T x' + 1)^d \tag{2.93}$$

donde d es un número natural, se añade 1 para que se incluyan términos cruzados con grados iguales o menores que d. En el caso en el que d sea igual a 1 entonces el núcleo es lineal más 1, sólo se tendría que ajustar el término de b en la función de decisión.

Para mostrar graficamente cada tipo de núcleo utilizaremos los datos que expuse en el ejemplo 4 donde utilizamos un núcleo radial ya que era el mejor para poder clasificar de mejor manera a los datos, a continuación mostraremos cómo es que se ve un núcleo polinomial de grado 2 con estos mismos datos (Figura 2.9).

```
> svmfit=svm(y~., data=dat[train ,], kernel ="polynomial
   ", cost=10, degree=2)
> summary(svmfit)

Call:
svm(formula = y ~ ., data = dat[train, ], kernel = "
   polynomial", cost = 10,
    degree = 2)


Parameters:
   SVM-Type:  C-classification
 SVM-kernel:  polynomial
       cost:  10
     degree:  2
```

```
      gamma:  0.5
    coef.0:  0

Number of Support Vectors:  37

 ( 18 19 )


Number of Classes:  2

Levels:
 -1 1
```

Como podemos ver este modelo arroja 37 vectores de soporte, 18 que pertenecen a la clase -1 y 19 que pertenecen a la clase $+1$, con los parámetros $C = 10$ y $\gamma = 2$, el núcleo que usamos fue el núcleo polinomial y el tipo de MSV fue el de clasificación (ya que también hay un tipo de MSV para regresión).

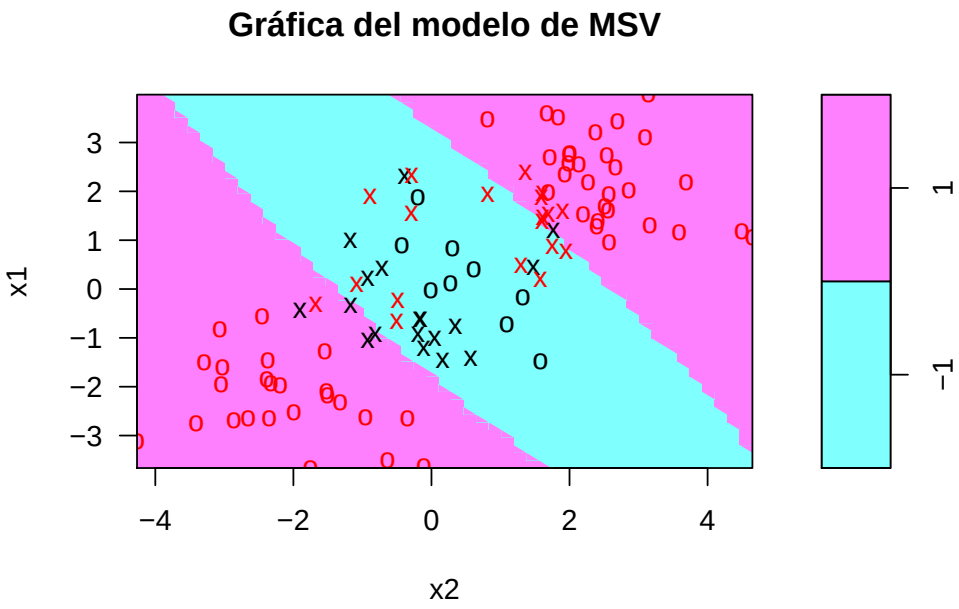


Figura 2.9: Ejemplo 4: núcleo polinomial de grado $d = 2$. Modelo de MSV con $C = 10$, $\gamma = 0.5$.

2.2.1. Ejemplo 3: conjunto linealmente no separable en una dimensión

De acuerdo a la figura 2.10, observamos que el caso es no separable linealmente, es por ello que usaremos un núcleo polinomial de grado 2 el cual está dado por la expresión (2.94), más detalles de los núcleos se ven en la sección 2.3.

$$k(x, x') = (x^T x' + 1)^2 \tag{2.94}$$

Dicho lo anterior, el problema dual está dado de la siguiente manera:

$$\text{maximizar } W(\alpha) \quad \alpha_1 + \alpha_2 + \alpha_3 - \left(2\alpha_1^2 + \frac{1}{2}\alpha_2^2 + 2\alpha_3^2\right) - \alpha_2(\alpha_1 + \alpha_3) \tag{2.95}$$

$$\text{sujeto a} \qquad \alpha_1 - \alpha_2 + \alpha_3 = 0 \tag{2.96}$$

$$0 \leq \alpha_i \leq C \quad \text{para} \quad i = 1, 2, 3. \tag{2.97}$$

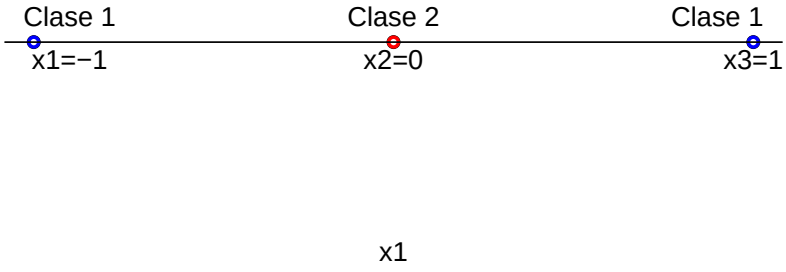


Figura 2.10: Ejemplo 3: datos no separables linealmente en una dimensión.

de (2.96) sabemos que $\alpha_2 = \alpha_1 + \alpha_3$, sustituyendo en (2.95), obtenemos lo siguiente:

$$W(\alpha) = 2\alpha_1 + 2\alpha_3 - (2\alpha_1^2 - \frac{1}{2}(\alpha_1 + \alpha_3)^2 + 2\alpha_3^2) \tag{2.98}$$

$$C \geq \alpha_i \geq 0 \text{ para } \ i = 1, 2, 3. \tag{2.99}$$

entonces:

$$\frac{\partial Q(\alpha)}{\partial \alpha_1} = 2 - \alpha_1 + \alpha_3 = 0 \tag{2.100}$$

$$\frac{\partial Q(\alpha)}{\partial \alpha_3} = 2 + \alpha_1 - 3\alpha_3 = 0 \tag{2.101}$$

$\alpha_1 = \alpha_3 = 1$. Para $C \geq 2$, la solución óptima es:

$$\alpha_1 = 1, \quad \alpha_2 = 2, \quad \alpha_3 = 1 \tag{2.102}$$

Por lo tanto para $C \geq 2$, $x = 1, x = 0$ y $x = -1$ son vectores de soporte. De (2.84) $b = -1$, la función de decisión nos queda de la siguiente manera:

$$D(x) = (x-1)^2 + (x+1)^2 - 3 = 2x^2 - 1 \tag{2.103}$$

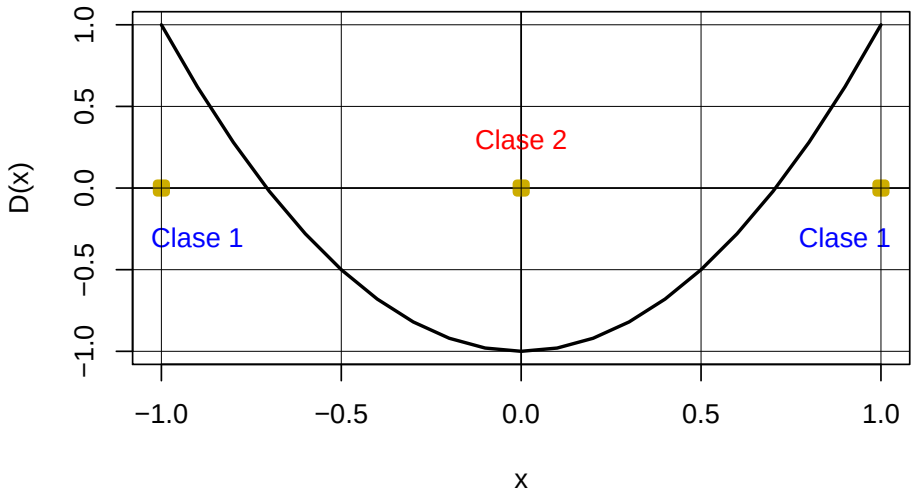


Figura 2.11: Ejemplo 3: separación de dos clases usando una función de decisión con núcleo polinomial de grado 2 (vectores de soporte señalados con cuadros amarillos).

En la figura 2.11 observamos cuáles son los vectores de soporte y la función de decisión que los separa (parábola remarcada de negro). Los límites de decisión, es decir, en donde corta la función de decisión al eje x, están dados por $x = \pm\frac{\sqrt{2}}{2}$ Por lo tanto, en el espacio de entrada el margen para la clase 2 (que es la distancia qu hay entre el dato de la clase 2 y los datos dela clase 1) es mayor que para la clase 1, aunque son iguales en el espacio de características (llevándolo a otra dimensión mayor).

Kernel Radial Basis Function núcleo radial ó núcleo Gaussiano

Es una función definida por la siguiente expresión:

$$K(x, x') = \exp(-\gamma\|x - x'\|^2) \tag{2.104}$$

donde γ es un parámetro positivo para controlar el radio, entonces (2.104) queda de la siguiente manera:

$$K(x, x') = \exp(-\gamma\|x\|^2)\exp(-\gamma\|x'\|^2)\exp(2\gamma x^T x') \tag{2.105}$$

debido a que:

$$\exp(2\gamma x^T x') = 1 + 2\gamma x^T x' + \frac{2\gamma^2(x^T x')^2}{2!} + \frac{(2\gamma)^3}{3!} + \ldots, \tag{2.106}$$

la ecuación (2.106) es el desarrollo de la serie de Taylor de la función exponencial y es la estandarización de una función núcleo, es por ello que se justifica su validez como una función núcleo.

De (2.78), tenemos que la función de decisión sería:

$$D(x) = \sum_{i \in S} \alpha_i y_i \exp(-\gamma\|x_i - x\|^2) + b \tag{2.107}$$

Aquí, los vectores de soporte son los centros de las funciones de base radial y debido a que los núcleo RBF utilizan la distancia euclidiana no son robustos a los valores atípicos.

Núcleo Mahalanobis

Normalmente las variables de entrada tienen una distribución de datos diferente, y esto puede afectar a la capacidad de generalización del clasificador, una forma de evitar este problema es normalizar los núcleo. Así como este método existen otros más que ayudan a evitar este problema, sin embargo nosotros nos enfocaremos en el núcleo Mahalanobis.

Primero explicamos la distancia de Mahalanobis entre una muestra de datos y el vector central de un *cluster*. Sea el conjunto de M datos, m-dimensionales

$\{x_1, \ldots, x_M\}$ para el *cluster*. Entonces el vector central y la matriz de covarianza de los datos se dan respectivamente por:

$$c = \frac{1}{M}\sum_{i=1}^{M} x_i \tag{2.108}$$

$$Q = \frac{1}{M}\sum_{i=1}^{M}(x_i - c)(x_i - c)^T \tag{2.109}$$

La distancia de Mahalanobis de x está dada por:

$$d(x) = \sqrt{(x-c)^T Q^{-1}(x-c)} \tag{2.110}$$

Debido a que la distancia de Mahalanobis es normalizada por la matriz de covarianza no necesitamos preocuparnos por las escalas de las variables de entrada. Otra característica interesante es que el promedio del cuadrado de las distancias de Mahalanobis para los datos de entrenamiento es m:

$$\frac{1}{M}\sum_{i=1}^{M}(x_i - c)^T Q^{-1}(x_i - c) = m \tag{2.111}$$

Entonces de acuerdo a la definción de la distancia de Mahalanobis, podemos definir a su núcleo de la siguiente manera:

$$K(x, x') = \exp(-(x - x')^T A(x - x')) \tag{2.112}$$

donde A es una matriz definitiva positiva, la distancia de Mahalanobis se calcula entre x y x', no entre x y c. El núcleo Mahalanobis es una extensión del núcleo RBF, es decir, estableciendo

$$A = \gamma I \tag{2.113}$$

donde $\gamma > 0$ es un parámetro para controlar el radio e I es la matriz unitaria de $m \times m$, obtenemos el núcleo RBF:

$$K(x, x') = \exp(-\gamma\|x - x'\|^2) \tag{2.114}$$

Para un problema binario, el núcleo Mahalanobis se utiliza para los datos pertenecientes a cualquiera de las dos clases. Suponiendo que $X = \{x_1, \ldots, x_M\}$ es el conjunto de todos los datos de entrenamiento, calculamos el centro y la matriz de covarianza por (2.108) y (2.109), respectivamente. Entonces nos aproximamos al núcleo Mahalanobis por:

$$K(x, x') = \left(-\frac{\delta}{m}(x - x')^T Q^{-1}(x - x')\right) \tag{2.115}$$

donde $\delta > 0$ es el factor para controlar la distancia de Mahalanobis. De (2.111), dividiendo el cuadrado de la distancia de Mahalanobis por m, la media del cuadrado de la distancia de Mahalanobis se normaliza a 1 independientemente del número de variables de entrada. Si los vectores de soporte están cerca de c, podemos esperar el efecto de normalización, y esto puede permitir limitar la búsqueda de un valor δ óptimo en un pequeño rango. Si usamos la matriz de covarianza completa involucraríamos una gran cantidad de tiempo si hay un alto número de variables de entrada, entonces consideramos dos casos: núcleos Mahalanobis con matrices diagonales de covarianza y núcleos Mahalanobis con matrices de covarianza completas. A los primeros se les llama núcleos diagonales de Mahalanobis y los últimos son llamados núcleos no-diagonales de Mahalanobis.

Para este tipo de núcleos necesitamos determinar los valores de δ y c. Pero dado que los núcleos Mahalanobis dados por (2.115) se determinan de acuerdo con la distribución de datos y se normalizan por m, el valor inicial de $\delta = 1$ es una buena selección. La selección del modelo es de la siguiente manera:

- Definir $\delta = 1$ y determinar el valor de c por validación cruzada, a esto se le conoce como primera etapa.
- Ajustando el valor de c determinado por la primera etapa, determinar el valor de δ por validación cruzada. Llamamos a esto segunda etapa.

Debido a que los núcleos Mahalonobis están normalizados por la matriz de covarianza, es invariante en escala. Por lo tanto, la transformación de escala de las variables de entrada no afecta el rendimiento de clasificación de la máquina de soporte vectorial.

Núcleo Sigmoidal

El núcleo sigmoidal es conocido también como núcleo tangente hiperbólico o como el núcleo de perceptron multicapa (MLP). El núcleo sigmoidal proviene del campo de redes neuronales, es interesante observar que un modelo MSV que utiliza una función de un núcleo sigmoidal es equivalente a una red neuronal perceptron de dos capas. Este núcleo es bastante popular para las máquinas de soporte vectorial debido a su origen en la teoría de redes neuronales. Además se ha encontrado que tiene un buen desempeño en la práctica.

Consideraremos el núcleo sigmoidal definido de la siguiente manera:

$$K(x, x') = tanh(\alpha(x^T x') + c) \quad \text{para} \quad \alpha, c \in R \tag{2.116}$$

Como vemos este núcleo toma dos parámetros, α que es la pendiente y la intersección, c.

Si $\alpha > 0$, podemos ver a α como un parámetro de escala de los datos de entrada, y a c como un parámetro de desplazamiento que controla el umbral del mapeo. Si $\alpha < 0$, el producto punto de los datos de entrada no sólo es escalado sino invertido, se concluye que $\alpha > 0$ y $c < 0$, es el caso más adecuado para el núcleo sigmoidal.

Como en los casos anteriores mostraremos con los datos del ejemplo 4 cómo es que se ve un núcleo sigmoidal (ver Figura 2.12), este modelo arroja 43 vectores de soporte, 21 que pertenecen a la clase -1 y 22 que pertenecen a la clase $+1$.

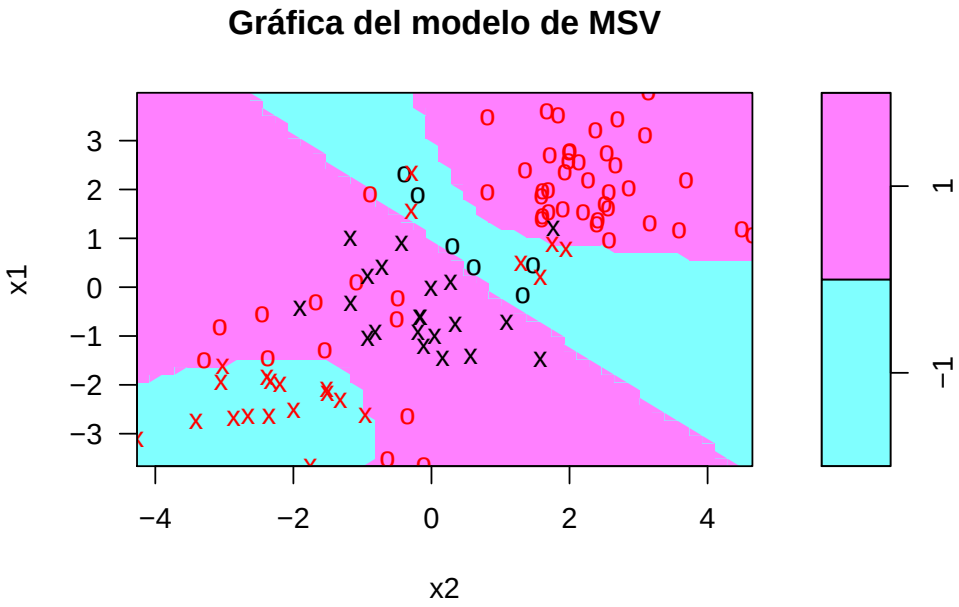


Figura 2.12: Ejemplo 4: núcleo sigmoidal. Modelo de MSV con $C = 10$ y $\alpha = 2$.

2.2.2. Ejemplo 4: núcleo radial con datos linealmente no separables

Como había mencionado antes, en la realidad es casi imposible poder encontrar un ejemplo con datos linealmente separables, es por ello que mostraremos un ejemplo de datos linealmente no separables. Aquí se utilizará el software `R` para mostrar el ejemplo no lineal con el núcleo radial, usaremos la paquetería `e1071`, la cual contiene implementaciones para una serie de métodos de aprendizaje estadístico. En particular, la función `svm`. Aquí se demuestra el uso de esta función en un ejemplo bidimensional para que podamos trazar el límite de decisión resultante. Primero generamos algunos datos con un límite de clase no lineal. La figura 2.13. La figura 2.13 muestra los casos de dos clases linealmente no separables, donde los puntos azules representan a la clase positiva (o clase 1) y los puntos rojos representan a la clase negativa (o clase 2).

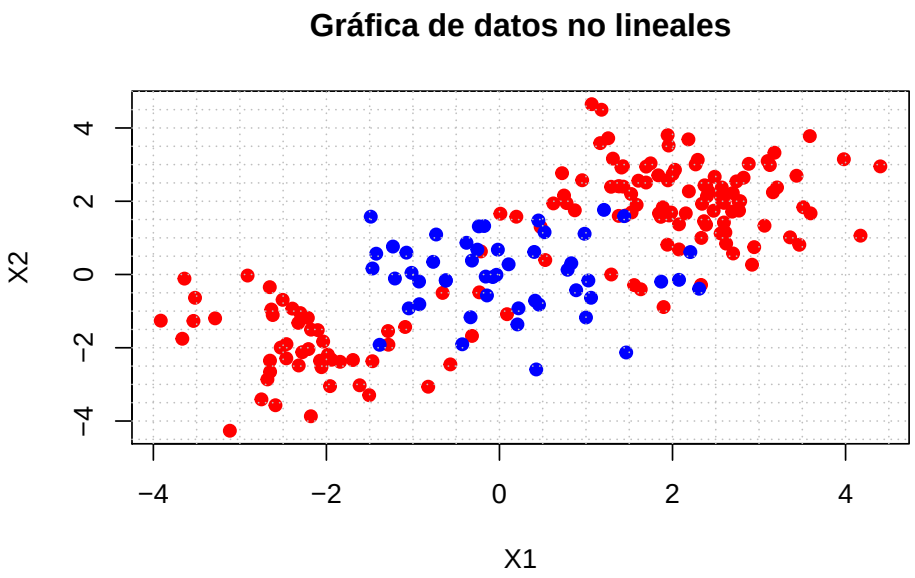


Figura 2.13: Ejemplo 4: datos linealmente no separables en dos dimensiones. Los puntos azules representan a la clase positiva y los puntos rojos a la clase negativa.

Los datos se dividen aleatoriamente en grupos de entrenamiento y prueba, lo que se debe hacer es ajustar los datos de entrenamiento utilizando la función `svm` $(\cdot)$ con un núcleo radial (ver sección 2.2) como se muestra en la figura 2.14.

En la figura 2.14 observamos un significativo número de errores de entrenamiento ya que al aplicar el modelo nos dio como resultado 37 vectores de soporte, 17 que pertenecen a la clase -1 y 20 a la clase $+1$. El número de vectores de soporte depende del valor del parámetro C como habíamos mencionado anteriormente, es decir, si aumentamos el valor del coste, podemos reducir el número de errores de entrenamiento. Sin embargo, debemos tener cuidado ya que esto produce un límite de decisión más irregular, como se muestra en la figura 2.15.

En el modelo que se muestra en la Figura 2.15 hay 26 vectores de soporte, 12 que pertenecen a la clase -1 y 14 de la clase $+1$, en este modelo los datos se ven mejor clasificados, sin embargo el valor de C es demasiado grande, es por ello que debemos aplicar la validación cruzada con la cual se seleccionarán los parámetros más adecuados para el modelo, es decir, γ y C. Los diferentes costos que se usaron fueron $C = 0.1, 1, 10, 100, 1000$ y $\gamma = 0.5, 1, 2, 3, 4, 5$, el mejor costo y γ que nos arrojó la técnica de validación cruzada fue el siguiente: $C = 1$ y $\gamma = 2$. A continuación se muestra los resultados que nos arroja la función `tune.svm`.

```
Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
    1     2

- best performance: 0.12

- Detailed performance results:
    cost gamma error dispersion
1  1e-01   0.5  0.27 0.11595018
2  1e+00   0.5  0.13 0.08232726
3  1e+01   0.5  0.15 0.07071068
4  1e+02   0.5  0.17 0.08232726
5  1e+03   0.5  0.21 0.09944289
6  1e-01   1.0  0.25 0.13540064
7  1e+00   1.0  0.13 0.08232726
8  1e+01   1.0  0.16 0.06992059
9  1e+02   1.0  0.20 0.09428090
10 1e+03   1.0  0.20 0.08164966
```

```
11 1e-01   2.0  0.25 0.12692955
12 1e+00   2.0  0.12 0.09189366
13 1e+01   2.0  0.17 0.09486833
14 1e+02   2.0  0.19 0.09944289
15 1e+03   2.0  0.20 0.09428090
16 1e-01   3.0  0.27 0.11595018
17 1e+00   3.0  0.13 0.09486833
18 1e+01   3.0  0.18 0.10327956
19 1e+02   3.0  0.21 0.08755950
20 1e+03   3.0  0.22 0.10327956
21 1e-01   4.0  0.27 0.11595018
22 1e+00   4.0  0.15 0.10801234
23 1e+01   4.0  0.18 0.11352924
24 1e+02   4.0  0.21 0.08755950
25 1e+03   4.0  0.24 0.10749677
```

La función `tune.svm` realiza una validación cruzada 10-fold (ver sección 1.6), la cual obtiene como mejores parámetros `cost=1` y $\gamma = 2$, con un valor de `best.performance=0.12`, es decir que al hacer distintas combinaciones entre el siguiente rango de parámetros `cost=c(0.1,1,10,100,1000)` y `gamma=c(0.5,1,2,3,4)` la que arrojó menos error fue $C = 1$ y $\gamma = 2$, a partir de esto podemos entonces tomar estos parámetros como óptimos.

En la Figura 2.14 podemos observar que los vectores de soporte que pertenecen a la clase 1 están marcados con una "x" de color negro y los vectores de soporte que pertenecen a la clase -1 están marcados con una "x" de color rojo.

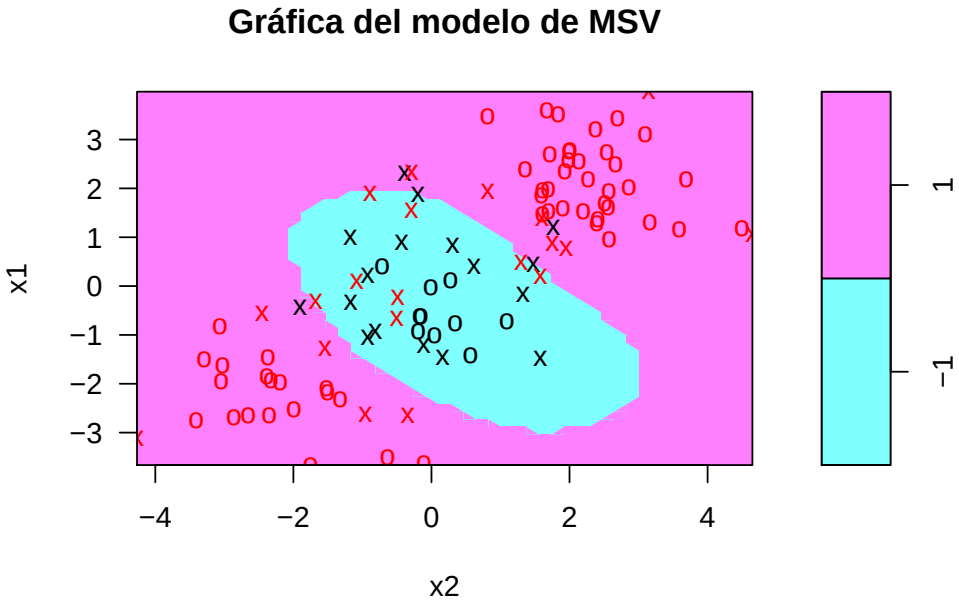


Figura 2.14: Ejemplo 4: núcleo radial. Modelo de MSV con $\gamma = 1$ y $C = 1$.

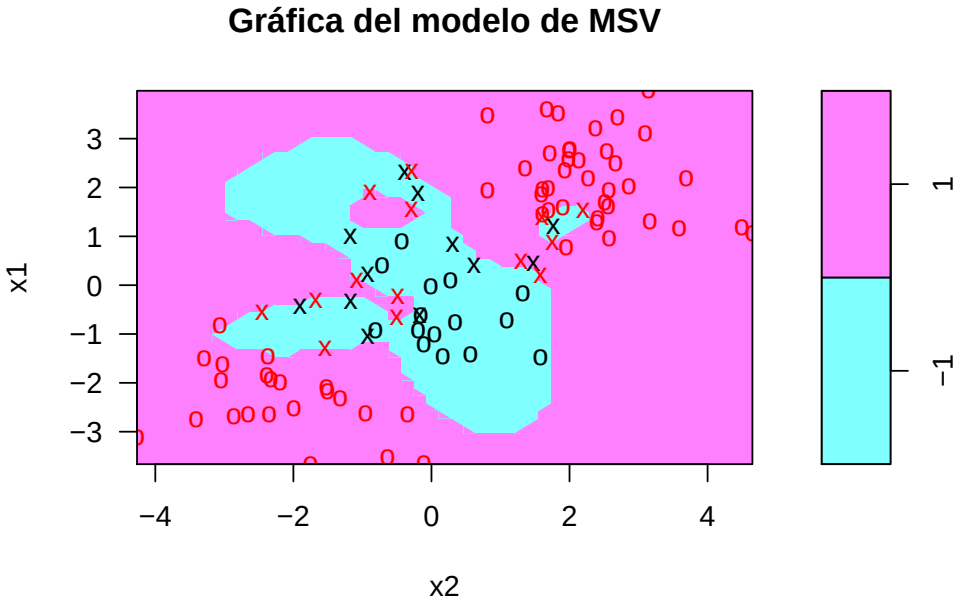


Figura 2.15: Ejemplo 4: núcleo radial. Modelo de MSV con $\gamma = 1$ y $C = 100000$.

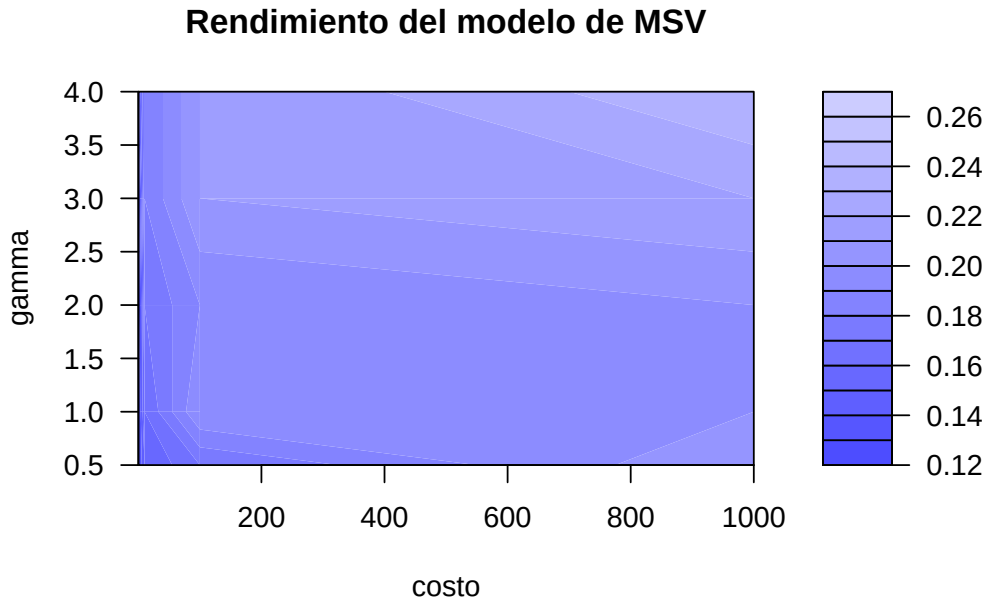


Figura 2.16: Ejemplo 4: rendimiento del modelo de MSV radial.

En la figura 2.16 se muestra el comportamiento de la función `tune.svm`, donde con azul fuerte se distinguen los valores óptimos de los parámetros para el modelo y conforme va bajando la tonalidad de azul los parámetros muestran más error.

Aplicando estos nuevos parámetros óptimos al modelo `svm` obtenemos el siguiente resultado que se muestra en la figura 2.17. Podemos ver que el margen de error en los datos a clasificar es menor con los nuevos parámetros. Los vectores de soporte que nos arroja este modelo son 43, 17 que pertencen a las clase -1 y 26 de la clase $+1$. Con este nuevo modelo sólo el 39 % de los datos de entrenamiento están mal clasificados:

```
    pred
true -1  1
  -1  5 18
  1  21 56
```

Para poder visualizar si en realidad es el mejor modelo, graficamos la curva ROC como se muestra en la figura 2.18.

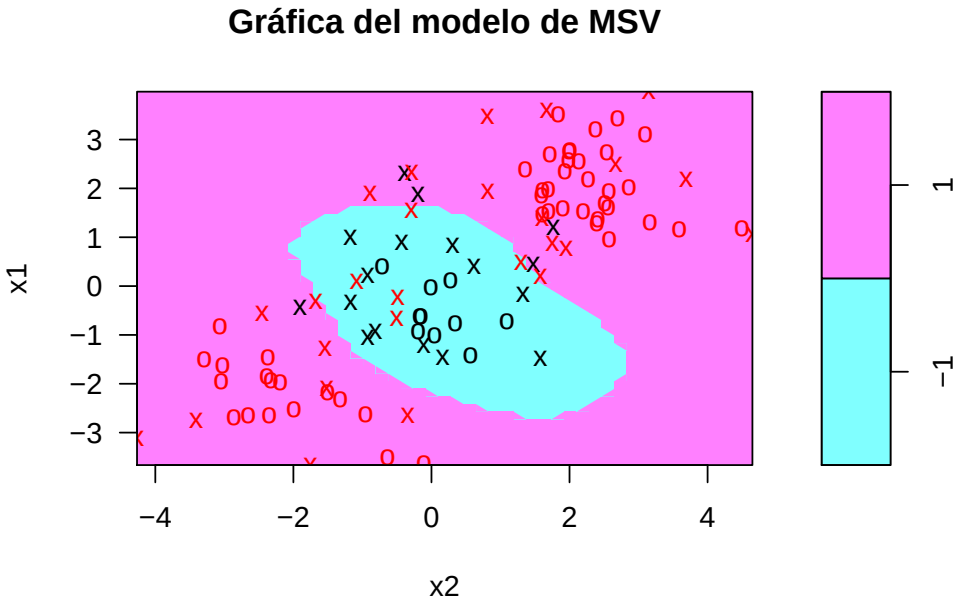


Figura 2.17: Ejemplo 4: núcleo radial. Modelo de MSV con $\gamma = 2$ y $C = 1$.

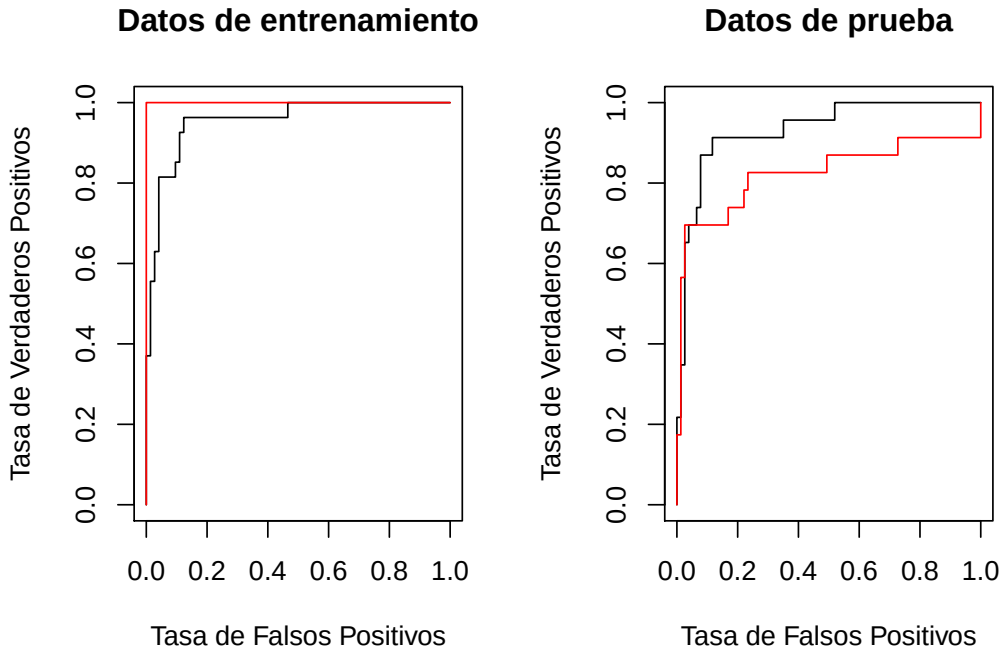


Figura 2.18: Ejemplo 4: núcleo radial. Curva ROC del modelo óptimo de MSV con $\gamma = 2$ y $C = 1$ (línea negra) y curva ROC óptima con $\gamma = 60$ y $C = 1$ (línea roja).

Como podemos ver en la gráfica 2.18, el MSV parece estar produciendo predicciones precisas. Al aumentar γ, podemos producir un ajuste más flexible y generar más mejoras en la precisión, sin embargo, estas curvas ROC están todas en los datos de entrenamiento. Estamos mas interesados en el nivel de precisión de predicción en los datos de prueba. Cuando calculamos las curvas ROC en los datos de prueba, el modelo con $\gamma = 2$ parece proporcionar los resultados mas precisos.

2.3. Máquinas de soporte vectorial para clasificación multiclase

Los problemas de reconocimiento de patrones multiclase (donde se tienen más de dos clases) suelen resolverse usando métodos basados en la combinación de muchas funciones de decisión de clasificación binaria. En esta sección nos enfocaremos en 4 métodos los cuales son:

- Máquinas de soporte vectorial uno contra todos (*one-against-all*).
- Máquinas de soporte vectorial por parejas (*pairwise*).
- Máquinas de soporte vectorial código de salida de corrección de errores (*error-correcting output code*, (ECOC)).
- Máquinas de soporte vectorial todos a la vez (*all-at-once*).

Máquinas de soporte vectorial uno contra todos (*one-against-all*)

En máquinas de soporte vectorial uno contra todos, lo que se hace es convertir un problema de clase n en n problemas de dos clases, y para el problema de dos clases i, la clase i se separa de las clases restantes, sin embargo por este tipo de formulación existen regiones no clasificables si usamos las funciones de decisión discretas. Para evitar este problema se deben tomar funciones de decisión continuas en lugar de funciones de decisión discretas y se resuelvan las regiones no clasificables.

Consideremos un problema de clase n, determinamos n funciones de decisión directas que separan una clase de las clases restantes, por lo tanto la i-ésima

función de decisión con el margen máximo que separa la clase i de las clases restantes está dada por:

$$D_i(x) = w_i^T \phi(x) + b \tag{2.117}$$

donde w_i es el vector l-dimensional, $\phi(x)$ es la función de mapeo que asigna x al espacio característico l-dimensional y b es el sesgo. Si $D_i(x) = 0$ entonces forma el hiperplano de separación óptimo.

Si los datos son separables, entonces los datos de entrenamiento pertenecientes a la clase i satisfacen $D_i(x) \geq 1$ y los pertenecientes a las clases restantes satisfacen $D_i(x) \leq -1$, los vectores de soporte satisfacen $|D_i(x)| = 1$.

Si los datos son no separables entonces los vectores de soporte *unbounded* satisfacen $|D_i(x)| = 1$ y los vectores de soporte *bounded* pertenecientes a la clase i satisfacen $D_i(x) \leq 1$ y los pertenecientes a una clase distinta de la clase i, $D_i(x) \geq -1$.

Si para el vector de entrada x, $D_i(x) > 0$ se satisface para una i, x se clasifica en la clase i. Si $D_i(x) > 0$ se satisface para varias $i's$ o si no hay i que lo satisfaga, x no es clasificable.

Máquinas de soporte vectorial por parejas (*pairwise*)

Aquí se determinan las funciones de decisión para todas las combinaciones de pares de clases, es decir, convierte el problema de n clases en $\frac{n(n-1)}{2}$ problemas de dos clases; para este método también existen regiones no clasificables. La función de decisión para la clase i contra la clase j, con el margen máximo es la siguiente:

$$D_{ij}(x) = w_{ij}^T \phi(x) + b_{ij} \tag{2.118}$$

donde w_{ij} es el vector l-dimensional, $\phi(x)$ es una función de mapeo que mapea x al espacio de características l-dimensional, b_{ij} es el sesgo y $D_{ij}(x) = -D_{ji}(x)$.

Las regiones:

$$R_i = \{x | D_{ij}(x) > 0, j = 1, \ldots, n, j \neq i\} \text{ para } \; i = 1, \ldots, n. \tag{2.119}$$

no se superponen, si x está en R_i entonces x pertenece a la clase i. Sin embargo x puede no estar en ninguna región de R_i, es por ello que se clasifica

a x por votación, es decir, para el vector de entrada x se calcula:

$$D_i(x) = \sum_{j \neq i, j, i=i}^{n} sign(D_{ij}(x)) \tag{2.120}$$

donde

$$sign(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ -1 & \text{si } x < 0 \end{cases} \tag{2.121}$$

entonces se clasifica a x en la clase:

$$\arg \max_{i=1,\ldots,n.} D_i(x) \tag{2.122}$$

Si x es clasificable en i entonces $D_i(x) = n-1$ y $D_k(x) < n-1$ para $k \neq i$.

Si x es no clasificable entonces $D_i(x)$ es distinto a $n-1$ y (2.122) puede ser satisfecha para diversas i's.

Máquinas de soporte vectorial código de salida de corrección de errores (*error-correcting output code*, ECOC).

Estos códigos de corrección de errores son los que detectan y corrigen errores en los canales de transmisión de datos, se utilizan para codificar resultados de clasificadores para mejorar la capacidad de generalización. Para las máquinas vectoriales de soporte, además de mejorar la generalización, pueden usarse para resolver regiones no clasificables.

Máquinas de soporte vectorial todos a la vez (*all-at-once*)

En esta sección se resuelven regiones no clasificables de problemas que son multiclase, se usan $(M \times n)$ variables de holgura. Para un problema de clase n se define a la función de decisión para la clase i por:

$$D_i(x) = w_i^T \phi(x) + b_i, \tag{2.123}$$

Para revisar más información sobre los dos modelos anteriores (*all-at-once* y *ECOC*), consultar (Shigeo, 2010).

2.3.1. Ejemplo 5: datos iris para clasificación multiclase

En esta sección utilizaremos la base de datos iris la cual es muy conocida y utilizada para la clasificación y el reconocimiento de patrones. Esta base de datos fue utilizada para demostrar las diferencias entre especies de plantas iris, contiene 3 clases de 50 casos cada una, donde cada clase se refiere a un tipo de planta iris: iris setosa, iris versicolor e iris virginica.

Los atributos son:

- Longitud del sépalo en cm (*Sepal.Length*).
- Ancho del sépalo en cm (*Sepal.Width*).
- Longitud del pétalo en cm (*Petal.Length*).
- Ancho del pétalo en cm (*Petal.Width*).
- Clase (*Species*).

La figura 2.19 muestra el diagrama de dispersión de las variables de la base de dato iris.Note que algunas de las variables están altamente correlacionadas.

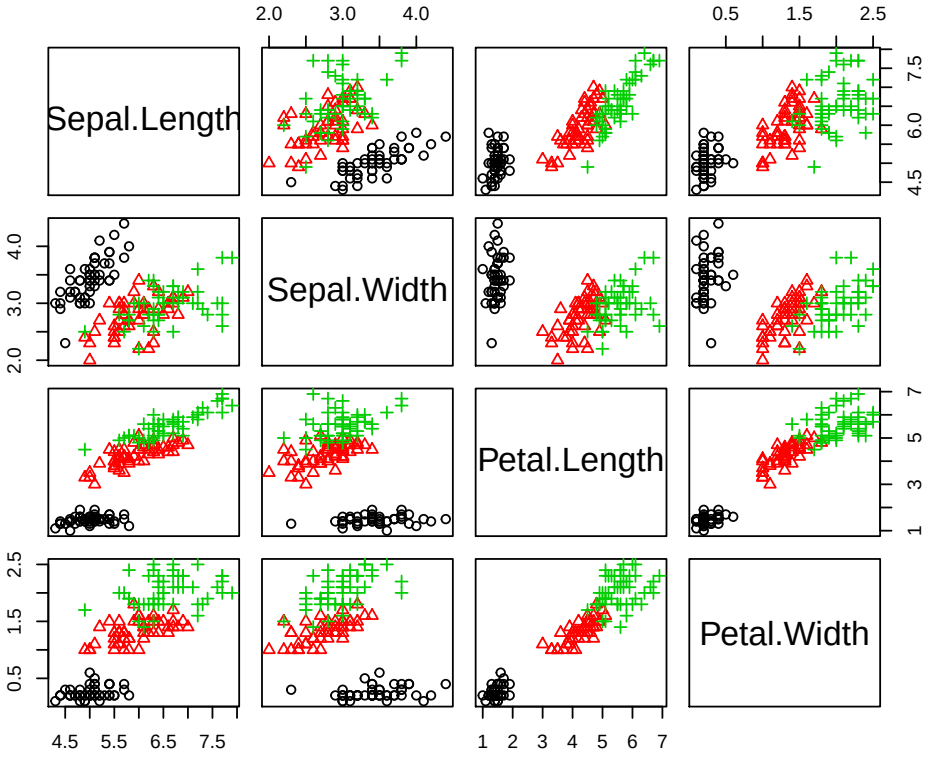


Figura 2.19: Iris: diagrama de dispersión.

Como mencionamos anteriormente esta base de datos se compone de 3 clases de tipo de plantas iris, dos de ellas no son separables linealmente:iris versicolor (triángulos rojos) e iris virginica (cruces verdes), y lo podemos ver graficamente en las figura 2.19.

Siguiendo lo que hemos dicho anteriormente debemos crear nuestros datos de entrenamiento (x) y los datos de prueba (y), en este caso x serían todos los atributos menos las especies y y serían las especies de iris.

Aplicando el modelo de máquinas de soporte vectorial obtenemos 38 vectores de soporte: 3 de la especie setosa, 19 de la especie versicolor y 16 de la especie virginica; con un núcleo de tipo RBF con $C = 5$ y $\gamma = 0.06$ (figura 2.20).

Los cálculos que a continuación se presentan se han realizado usando el 100 % de los datos como datos de entrenamiento, y también se ha usado el 100 % de los datos como datos de prueba, sin embargo, se podría dividir el conjunto

de datos usando un porcentaje α de los datos como datos de entrenamiento, digamos el 75 %, y el resto de los datos $1-\alpha$ como datos de prueba, es decir el 25 % de los datos, otra opción sería usando una validación cruzada k-veces como la que se menciona en la sección 1.6.

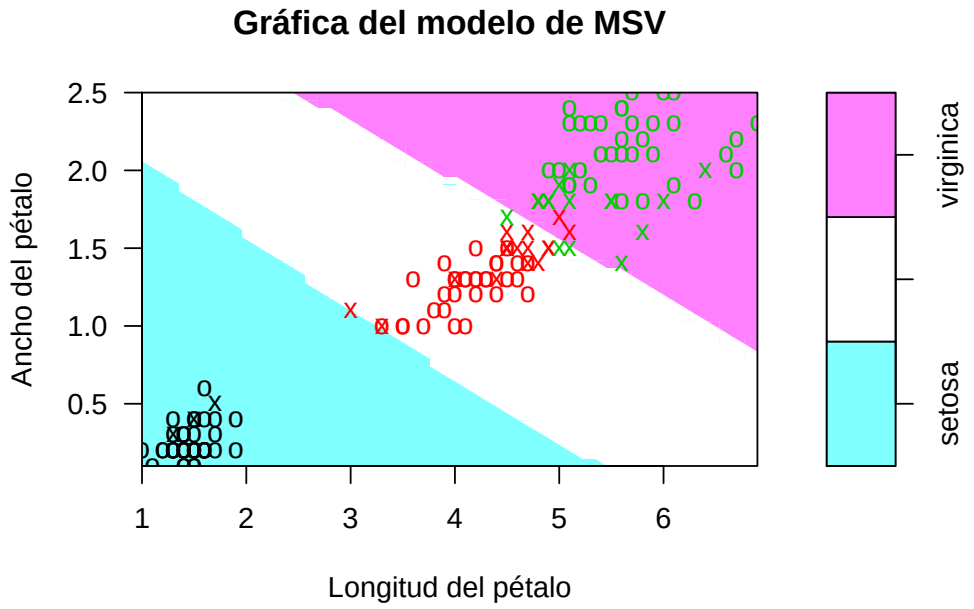


Figura 2.20: Iris: núcleo radial. Modelo de MSV con $C = 5$ y $\gamma = 0.06$.

```
> modelofinal <- svm(Species ~ ., data = iris, method =
   "C-classification", kernel = "radial", cost = 5,
   gamma = 0.06)
> summary(model)


Call:
svm(formula = Species ~ ., data = iris, method = "C-
   classification",
    kernel = "radial", cost = 5, gamma = 0.06)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  5
      gamma:  0.06
```

```
Number of Support Vectors:  38

 ( 3 19 16 )


Number of Classes:  3

Levels:
 setosa versicolor virginica
```

Para poder escoger los parámetros que son los más adecuados para el modelo lo que se hace es usar la función `tune.svm`, (figura 2.21) la cual permite seleccionar los valores óptimos de los parámetros de la función núcleo elegida y del costo.

```
> objfinal<- tune.svm(Species~., data = iris, gamma =c
   (1:30)/150, cost=c(1:100)/5)
> summary(objfinal)

Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:
 gamma cost
  0.06    5

- best performance: 0.01333333

- Detailed performance results:
           gamma cost      error dispersion
1    0.006666667    1 0.14000000 0.07981460
2    0.013333333    1 0.09333333 0.07166451
3    0.020000000    1 0.06000000 0.04919099
4    0.026666667    1 0.04666667 0.05488484
5    0.033333333    1 0.04666667 0.05488484
6    0.040000000    1 0.04666667 0.05488484
7    0.046666667    1 0.05333333 0.05258738
8    0.053333333    1 0.05333333 0.05258738
9    0.060000000    1 0.04666667 0.05488484
10   0.066666667    1 0.04666667 0.05488484
```

```
11   0.073333333    1 0.04666667 0.05488484
12   0.080000000    1 0.04666667 0.05488484
13   0.086666667    1 0.04666667 0.05488484
14   0.093333333    1 0.04666667 0.05488484
15   0.100000000    1 0.04000000 0.04661373
16   0.106666667    1 0.03333333 0.04714045
17   0.113333333    1 0.02666667 0.03442652
18   0.120000000    1 0.02666667 0.03442652
19   0.126666667    1 0.03333333 0.03513642
20   0.133333333    1 0.03333333 0.03513642
```

En la función se omitieron las demás combinaciones ya que son más de 3000 las que realizó el programa y sólo se pusieron las primeras 20. Con este resultado podemos observar cómo la función `tune.svm` realiza validaciones cruzadas con los distintos parámetros que metimos en la misma función y hace combinaciones para poder escoger los valores de los parámetros óptimos para el modelo. En este caso los valores óptimos, es decir los que minimizan el error (error= 0.01333333) y permiten clasificar de mejor manera estos datos son $\gamma = 0.06$ y $C = 5$. Y lo podemos observar gráficamente en la figura 2.21.

Anteriormente usamos un núcleo radial ya que es el más adecuado para clasificar con menor error a las 3 especies, sin embargo, podemos probar con otro tipo de núcleo. Mostraremos cómo es que cambia la clasificación con distintos núcleo. Empezaremos con el núcleo lineal (figura 2.22), este modelo nos da 19 vectores de soporte: 2 de la especie setosa, 9 de la especie versicolor y 8 de la especie virginica; con $C = 5.8$ seleccionado con la función `tune.svm` (ver figura 2.23). Como podemos ver este modelo tiene menor cantidad de vectores de soporte, sin embargo hay un número mayor de datos que no están correctamente clasificados, es por ello que aún es preferible el núcleo radial.

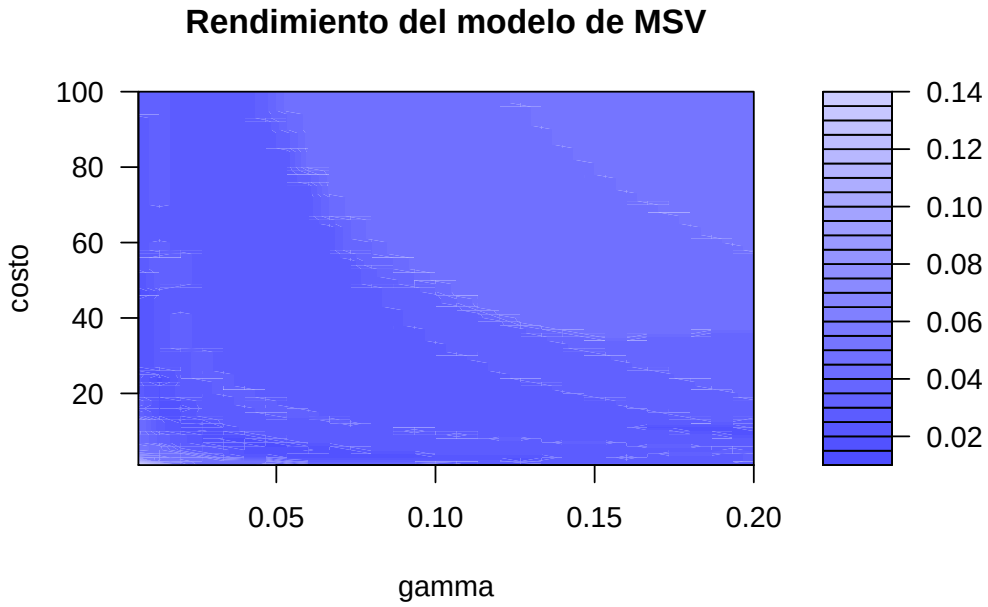


Figura 2.21: Iris: rendimiento del modelo. Comportamiento de la función `tune.svm`.

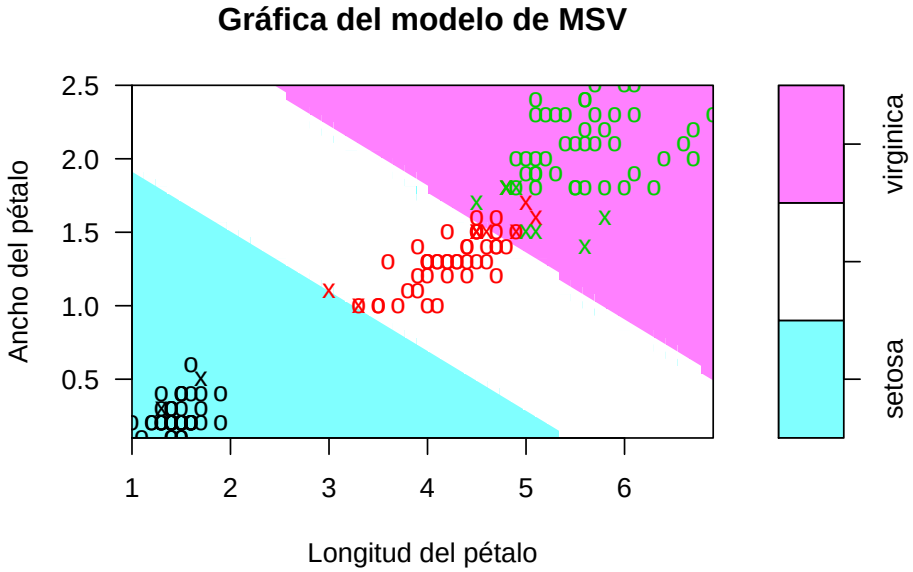


Figura 2.22: Iris: núcleo lineal. Modelo de MSV con $C = 5.3$.

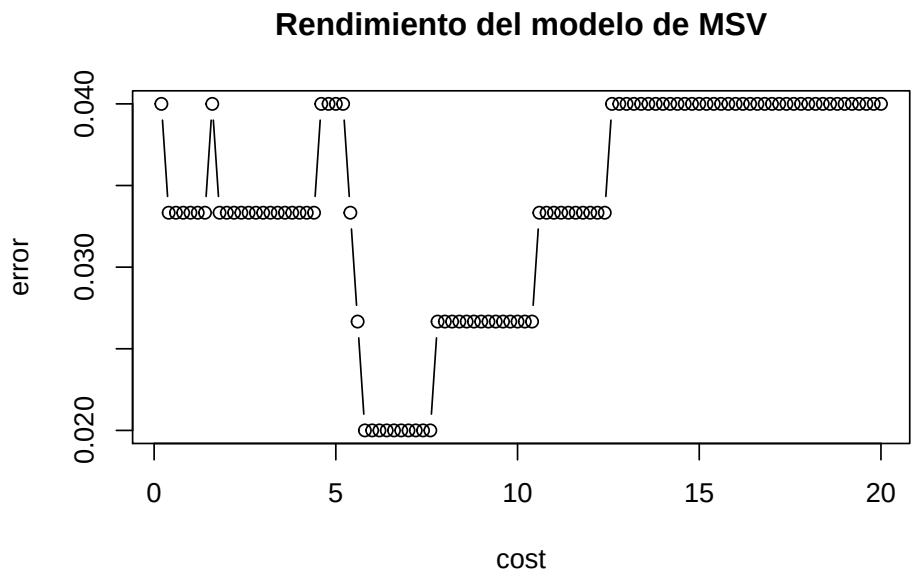


Figura 2.23: Iris: núcleo lineal. Rendimiento del modelo.

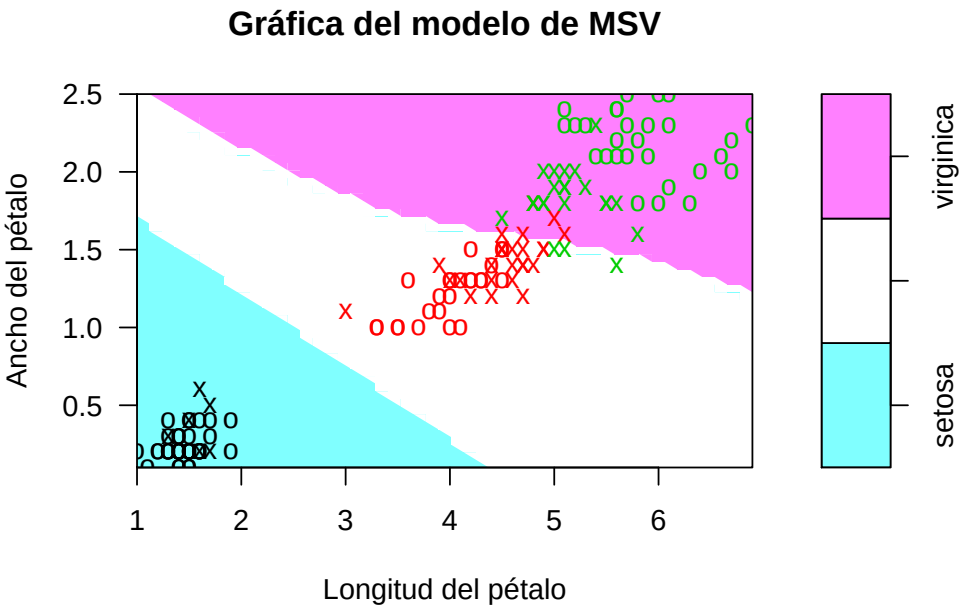


Figura 2.24: Iris: núcleo polinomial. Modelo de MSV con $C = 1$ y $d = 3$.

Utilizando un núcleo polinomial (figura 2.24) da como resultado 54 vectores

de soporte: 6 de la especie setosa, 26 de la especie versicolor y 22 de la especie virginica. Este modelo tiene mayor cantidad de vectores de soporte a comparación del modelo con el núcleo radial; sin embargo, al igual que en la gráfica anterior se puede observar que hay un número mayor de datos que no están correctamente clasificados.

Por último con el núcleo sigmoidal tenemos 44 vectores de soporte: 3 de la especie setosa, 22 de la especie versicolor y 19 de la especie virginica (fig. 2.24), como podemos ver el mejor modelo es con un núcleo radial, ya que por cada modelo no son tan eficientes como los arrojados con el modelo usando un núcleo radial.

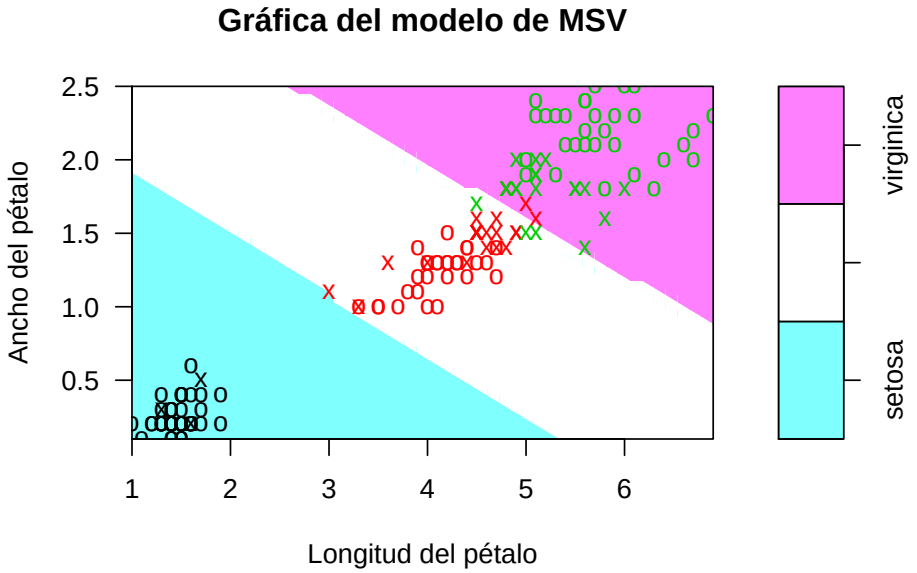


Figura 2.25: Iris: núcleo sigmoidal. Modelo de MSV con $C = 4.75$ y $\alpha = 0.06$.

Predicción

Al hacer la predicción para poder notar cuáles datos son correctamente clasificados y cuáles no, obtenemos lo siguiente. Vemos que con el tipo de núcleo que utilizamos (núcleo radial), sólo uno especie que es versicolor se estaría clasificando mal en la especie virginica y uno que es de la especie virginica se estaría clasificando mal en la especie versicolor (tabla 2.2), y por lo tanto sólo 2 datos están mal clasificados.

y / predicción	setosa	versicolor	virginica
setosa	50	0	0
versicolor	0	49	1
virginica	0	1	49

Tabla 2.2: Iris: núcleo radial. Tabla de predicción.

Haciendo la tabla de predicción para el núcleo lineal, vemos que se estarían clasificando mal solamente 4 plantas de la especie versicolor en la especie virginica (tabla 2.3), es decir, en este modelo habría más error que usando un núcleo radial ya que se estarían clasificando 4 datos erróneamente en una clase que no les corresponde.

y / predicción	setosa	versicolor	virginica
setosa	50	0	0
versicolor	0	46	0
virginica	0	4	50

Tabla 2.3: Iris: núcleo lineal. Tabla de predicción.

Usando un núcleo polinomial se estarían clasificando mal 7 datos de la especie virginica en la especie versicolor, sin embargo, las clases setosa y versicolor estarían correctamente clasificadas (tabla 2.4).

y / predicción	setosa	versicolor	virginica
setosa	50	0	0
versicolor	0	50	7
virginica	0	0	43

Tabla 2.4: Iris: núcleo polinomial. Tabla de predicción.

Con un núcleo sigmoidal se estaría clasificando mal un dato de la especie virginica en la especie versicolor, 4 datos de la especie versicolor en la especie virginica y en total se clasificarían mal 5 datos (tabla 2.5).

y / predicción	setosa	versicolor	virginica
setosa	50	0	0
versicolor	0	46	1
virginica	0	4	49

Tabla 2.5: Iris: núcleo sigmoidal. Tabla de predicción.

2.4. Dimensión VC

La Dimensión VC (Vapnik-Chervonenkys) es la base teórica de las máquinas de soporte vectorial (MSV). Se utiliza en todos los resultados importantes de la teoría estadística del aprendizaje, mide la capacidad o complejidad de la máquina al realizar algún algoritmo de clasificación.

Sea un modelo de clasificación $F = f(x, w)$, donde w son los parámetros libres adaptables a la máquina y x es algún dato a clasificar, la Dimensión VC (h) es la cardinalidad del mayor conjunto de puntos de entrenamiento organizados que el modelo F puede separar correctamente. En el caso de la clasificación binaria este conjunto se compone por funciones indicadoras definidas como $\phi_F(x, w) \in \{0, 1\}$ ó $\phi_F(x, w) \in \{-1, 1\}$, para toda x, w. Para la clasificación en MSV se usa la segunda forma para facilitar los cálculos algebraicos del problema. Estas funciones indican hiperplanos orientados, algebraicamente un hiperplano orientado está definido en R^n como:

$$\phi_F(x, w) = sign(u(x)) = \begin{cases} 1 & \text{si } u \geq 0 \\ -1 & \text{si } u < 0 \end{cases} \tag{2.124}$$

donde $u(x) = w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n$.

Por ejemplo, ¿podemos encontrar $f(x, w) = w_0 + w_1x_1 + w_2x_2$ que pueda separar estos puntos? (Ver la figura 2.26 que gráfica dos puntos en dos dimensiones).

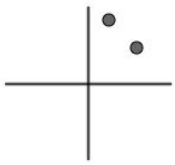


Figura 2.26: Dos puntos en dos dimensiones.

La respuesta es sí ya que hay 4 igual a (2^2) posibles conjuntos de entrenamiento. Como se muestra en la figura 2.27,las formas de separarlos serán (gráficas de izquierda a derecha): Que un punto esté en el grupo de los positivos y el otro punto en el grupo de los negativos(primera gráfica), o viceversa (segunda gráfica), o que ambos puntos estén en el grupo de los negativos (tercera gráfica), o ambos en los positivos (cuarta gráfica).

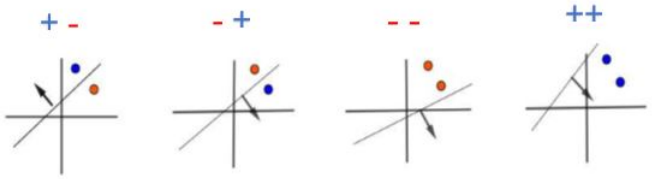


Figura 2.27: Separación de dos puntos en dos dimensiones.

Este mismo procedimiento se puede aplicar con 3 puntos, como se muestra en la figura 2.28.

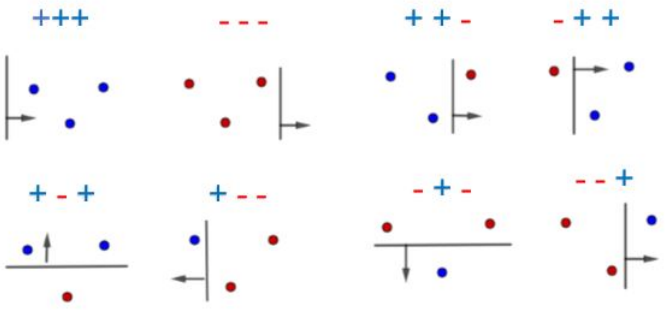


Figura 2.28: Separación de 3 puntos en dos dimensiones.

Un conjunto de h puntos puede ser clasificado de 2^h formas posibles y por

cada caso puede encontrarse al menos un hiperplano orientado que clasifique correctamente a los puntos. Supongamos que tenemos un conjunto de rectas orientadas en dos dimensiones, $u(x) = w_0 + w_1x_1 + w_2x_2$ y usamos estas funciones para separar 3 puntos en las $2^3 = 8$ maneras posibles de hacerlo obtendríamos lo que se observa en la figura 2.28. En este caso la dimensión VC para este conjunto en R^2 es igual a $n + 1 = 2 + 1 = 3$.

2.5. Minimización del riesgo estructural

La mayoría de los métodos de aprendizaje se centran en minimizar los errores cometidos por el modelo generado a partir de los ejemplos de entrenamiento (error empírico), las MSV tratan de minimizar el riesgo estructural (*Structural Risk Minimization*, SRM).

En general la esperanza del riesgo esperado está dada por:

$$R(w) = \int \frac{1}{2}|y - f(x, \alpha)|dP(x, y) \tag{2.125}$$

donde $|y - f(x, \alpha)|$ es una cantidad llamada pérdida, la cual sólo puede tomar los valores 0 y 1. La función $P(x, y)$ es la función de distribución de probabilidad desconocida, donde los datos son independientes identicamente distribuidos (iid). A la cantidad $R(w)$ se le conoce como el riesgo esperado. El riesgo empírico se define como la tasa media de error en el conjunto de entrenamiento (para un número fijo o número finito de observaciones) y está definido de la siguiente manera:

$$R_{emp}(w) = \frac{1}{2l}\sum_{i=1}^{l}|y - f(x, \alpha)| \tag{2.126}$$

en la ecuación 2.126 a diferencia de la ecuación 2.125 no aparece la función de distribución de probabilidad. El R_{emp} es un número fijo para una elección particular de w y para un conjunto de entrenamiento particular.

El riesgo estructural se controla por medio de dos parámetros:

1. La capacidad de la máquina, es decir la dimensión VC.

2. La reducción del error en el conjunto de entrenamiento.

La SRM controla la generalización de la máquina de soporte vectorial la cual ha sido propuesta por Vapnik (Vapnik, 1998) y se refiere a la capacidad de la máquina para clasificar correctamente datos con los cuales no ha sido entrenada, es decir, es capaz de predecir el resultado correcto para nuevos datos de entrenamiento. Para implementar el principio SRM se necesita una estructura anidada de espacios de hipótesis:

$$S_1 \subset S_2 \subset ... \subset S_n \subset ... \subset S \tag{2.127}$$

donde $S_n = \{f(x, w); w \in \Lambda_n\}$, donde Λ es el conjunto de pesos de la máquina, con dimensiones VC:

$$h_1 \leq h_2 \leq ... \leq h_n \leq ... \leq h \tag{2.128}$$

En tal conjunto anidado de funciones, cada función siempre contiene un función previa menos compleja, por ejemplo polinomios de orden creciente. Para poder trabajar con cantidades no limitadas de datos, Vapnik estableció el principio de SRM, en el cual para alguna $w \in \Lambda$ y $l > h$, donde l es el número de muestras y h es la dimensión VC, con probabilidad $1 - \eta$, se cumple la siguiente cota:

$$R(w) \leq R_{emp}(w) + \sqrt{\frac{h\left(\log\frac{2l}{h} + 1\right) - \log\frac{n}{4}}{l}} \tag{2.129}$$

Ambos lados de la ecuación 2.129 deben ser pequeños, es decir, tanto la dimensión de VC como el riesgo empírico deben minimizarse al mismo tiempo. Entonces a partir de la ecuación 2.129 podemos decir que el principio de SRM se basa en resolver el siguiente problema:

$$\min_{S_n} \left(R_{emp}(w) + \sqrt{\frac{h\left(\log\frac{2l}{h} + 1\right) - \log\frac{n}{4}}{l}} \right) \tag{2.130}$$

El principio SRM puede ser difícil de implementar por las siguientes razones:

- La dimensión VC de S_n puede ser difícil de calcular, ya que sólo hay un pequeño número de modelos para los cuales sabemos cómo calcular la dimensión VC, aparte de que es difícil de calcular en clasificadores arbitrarios.

- Suponiendo que se pueda calcular la dimensión VC de S_n, aún así no sería fácil resolver el problema de (2.130).

Por lo consiguiente implementar este principio no es sencillo, debido al cálculo de la dimensión VC. Las MSV logran este objetivo ya que minimizan un límite en la dimensión VC y el número de errores de entrenamiento al mismo tiempo.

Capítulo 3

Aplicación

3.1. Resumen y Resultados

Para la información de este capítulo me basé en los siguientes artículos (Pérez et al., 2016) y (Naranjo et al., 2016). Se basa inicialmente en 44 características, extraídas de grabaciones de vocales sostenidas (que la persona pronuncie por 3 segundos una vocal sin interrupciones) de 80 sujetos. El número de características se ha reducido aún más mediante la selección de caracterśticas. La base de datos está compuesta por 40 sujetos control y 40 sujetos con EP pertenecientes a diferentes estados de gravedad de la enfermedad y bajo tratamiento prescrito. Se promediaron mediciones repetidas por individuo antes de asignarse al sujeto, evitando la práctica habitual de considerar mediciones dentro del mismo sujeto como independientes. En este caso se tomó el promedio de las mediciones, sin embargo se pudo haber calculado la moda o la mediana, no se hizo uso de alguna regla en específico para obtener las mediciones correspondientes.

En este capítulo se usará el modelo de máquinas de soporte vectorial para poder discriminar automáticamente a las personas sanas de las personas con enfermedad de Parkinson (EP) a partir de las grabaciones de voz. Para esto se usará una base de 80 sujetos, se ajustarán distintos tipos de núcleos (lineal, polinomial, radial y sigmoidal) para poder ver la eficacia de cada uno de ellos al momento de discriminar a los pacientes con EP de los que no tienen EP y se elegirá el que arroje el resultado más óptimo para este estudio. Se realizarán dos tipos de análisis debido a la alta correlación entre los grupos

de variables: uno que incluye todas las 44 características del estudio y otro en el que se escogieron sólo 11 características, esto para ver que tan similares son los resultados y poder escoger así el mejor modelo. Se espera que el modelo de 11 características y el núcleo lineal sean los que arrojen el mejor resultado ya que por un lado existe una alta correlación entre las variables y el resultado puede ser muy similar al resultado que arrojaría el modelo con las 44 características y un modelo con menos variables es más eficiente; y por otro lado debido a que sólo tenemos dos conjuntos de datos y estamos manejando una sola dimensión el núcleo que más se adaptaría sería el lineal. Para poder llegar a este resultado se utilizó el método de la validación cruzada, es decir se repetió el modelo 100 veces con los dos tipos de análisis (modelo con 44 y 11 características), esto para poder probar la eficacia y el comportamiento que tenían las máquinas de soporte vectorial al hacerlas consecutivamente y así obtener un resultado óptimo.

3.2. Enfermedad de Parkinson

La enfermedad de Parkinson (EP) es la segunda enfermedad neurodegenerativa más frecuente, es un trastorno neurológico progresivo causado por la muerte o pérdida de las neuronas dopaminérgicas que desempeñan un papel clave en la coordinación del movimiento a nivel de tono muscular en diferentes áreas del sistema nervioso. La media de edad del comienzo de esta enfermedad está en torno a los 60 años, el mayor riesgo para padecer la enfermedad es la edad avanzada.

La voz y el habla, dependientes de las funciones laríngeas, respiratorias y articulatorias, también se ven afectados en personas con EP. Las señales no motoras también pueden afectar el lenguaje, la cognición y el estado de ánimo, lo que puede afectar a la comunicación. El deterioro vocal es muy probablemente uno de los primeros signos de la enfermedad, desde las primeras etapas de la EP hay anormalidades sutiles en el habla que pueden no ser perceptibles para los oyentes, pero pueden ser evaluadas de manera objetiva realizando análisis acústicos en señales de habla registradas.

La *disartria hipocinética* es comúnmente uno de los primeros síntomas de la EP, ésta es un desorden del habla caracterizado por problemas con la articulación normal del habla y un cierto retraso o torpeza al hablar. La *disfonía* es otro tipo de trastorno que sufren las personas con EP, produce síntomas tales como reducción de la sonoridad, dificultad respiratoria, torpeza, disminución

de la energía y un temblor exagerado al hablar.

Dado que el 90 % de las personas con EP padecen trastornos del habla, los cambios producidos por la enfermedad pueden ser medidos acústicamente y así diagnosticar la enfermedad. Debido a esto algunos investigadores de la EP han desarrollado modelos capaces de discriminar entre un individuo sano de uno que tiene EP, a través de datos que provienen de grabaciones de voz en los que la persona realiza fonaciones sostenidas con la vocal /a/ lo más estables que se les sea posible durante al menos 5 segundos. El desarrollo de sistemas remotos precisos considerando características extraídas de grabaciones de voz puede ser muy útil para ayudar a diagnosticar la EP en sus primeras etapas.

Una cuestión importante que tratar, tanto en el diagnóstico como en las aplicaciones de seguimiento, es la elección de un tratamiento estadístico adecuado. En este contexto, se ha vuelto habitual en la literatura aplicar métodos de clasificación.

3.3. Metodología

3.3.1. Participantes

Se consideró a 80 sujetos mayores de 50 años los cuales realizaban grabaciones de voz y seguían un protocolo de encuesta. Estaban clasificados como se muestra en la tabla 3.1.

	Hombres	Mujeres	Media de edad (± desviación estándar) en años
Sin EP (0)	22	18	66.38 ± 8.38
Con EP (1)	27	13	69.58 ± 7.82

Tabla 3.1: Parkinson, clasificación de los participantes en el estudio.

Ninguna de las personas del grupo de control tiene antecedentes de síntomas relacionados con la EP o cualquier otro tipo de síndrome de trastorno del movimiento. Los pacientes con EP presentaban al menos dos de los siguientes síntomas: temblor al reposo, bradicinesia o rigidez muscular. Las personas que participaron en el estudio y que tenían EP son miembros de la Asociación Regional para la Enfermedad de Parkinson en Extremadura (España).

3.3.2. Grabaciones de voz

Las grabaciones estaban basadas en que las personas hicieran la tarea vocal de la fonación sostenida de la vocal /a/ con un tono y una sonoridad cómoda, lo más constante posible, esta fonación tuvo que mantenerse durante al menos 5 segundos. Producir unos segundos sostenidos /a/ duraderos da lugar a una señal a partir de la cual se pueden extraer parámetros acústicos. La tarea se repitió tres veces por individuo, sin embargo no se usaron las grabaciones repetidas; se sacó el promedio de las 3 grabaciones por cada individuo. Se hizo el promedio de las 3 grabaciones porque existe una variabilidad biológica, es decir, cada grabación a pesar de que las emitiera la misma persona no son idénticas unas a otras, son reconocibles como pertenecientes al mismo individuo sin embargo pueden existir diferencias en cada grabación.

Los datos de voz se registraron utilizando un ordenador portátil con una tarjeta de sonido externa (TASCAM US322) y un micrófono de diadema (AKG520). La grabación digital se realizó a una frecuencia de muestreo de 44.1KHz y a una resolución de 16 bits/muestra utilizando el software Audacity (versión 2.0.5).

3.3.3. Extracción de características

La base de datos está compuesta por 44 características acústicas, las cuales están clasificadas en 5 grupos:

- Jitter (medidas de perturbación del tono local o frecuencia).
- Shimmer (medidas de la perturbación de la amplitud).
- Características del ruido (proporción de señales a ruido, HNR).
- Mediciones de las frecuencias cepstrales de Mel (MFCC y Delta).
- Mediciones no lineales.

La composición de los grupos anteriores se mostrará en las tablas 3.2, 3.3, 3.4 y 3.5.

Las características del ruido, que son la proporción de armonías en los ruidos (HNR, por sus siglas en inglés) es una medición del relativo nivel del ruido

Característica	Descripción
Jitter_rel	Perturbación relativa del tono local (expresada en porcentaje).
Jitter_abs	Perturbación absoluta del tono local.
Jitter_RAP	Perturbación relativa promedio del tono local.
Jitter_PPQ	Cociente de perturbación del tono local.

Tabla 3.2: Parkinson, medidas de perturbación del tono local (Jitter).

Característica	Descripción
Shimmer_loc	Perturbación local de la amplitud.
Shimmer_dB	Perturbación de la amplitud en decibeles.
Shimmer_APQ3	Cociente de perturbación de 3 puntos de amplitud.
Shimmer_APQ5	Cociente de perturbación de 5 puntos de amplitud.
Shimmer_APQ11	Cociente de perturbación de 11 puntos de amplitud.

Tabla 3.3: Parkinson, medidas de perturbación de la amplitud (Shimmer).

Característica	Descripción
HNR05	Proporción de armonías a ruidos entre 0 y 500 Hz.
HNR15	Proporción de armonías a ruidos entre 0 y 1500 Hz.
HNR25	Proporción de armonías a ruidos entre 0 y 2500 Hz.
HNR35	Proporción de armonías a ruidos entre 0 y 3500 Hz.
HNR38	Proporción de armonías a ruidos entre 0 y 3800 Hz.

Tabla 3.4: Parkinson, características del ruido (HNR).

Característica	Descripción
RPDE	Entriopía de la densidad del periodo de recurrencia.
DFA	Análisis de fluctuación sin tendencias.
PPE	Entropía del periodo del tono.
GNE	Cociente de excitación glotal sobre el ruido.

Tabla 3.5: Parkinson, características de mediciones no lineales.

en el habla, existen varios tipos de acuerdo al tiempo o la frecuencia; en este caso se han obtenido de acuerdo a intervalos de distintas frecuencias (tabla 3.4).

La tabla 3.5 muestra las mediciones no lineales de las características de voz. La RPDE estima la duración de los ciclos de las cuerdas vocales en el tiempo

transcurrido entre colisiones de cuerdas vocales contiguas. La PPE es una log-transformación de la frecuencia fundamental, la cual sirve como una medición de la disfonía. El GNE cuantifica la cantidad de excitación vocal causada por la oscilación de las cuerdas vocales contra la excitación producida por el ruido de turbulencia.

Los coeficientes cepstrales de frecuencias de Mel (MFCC, por sus siglas en inglés) están relacionadas con el espectro del habla, detectan pequeños cambios en la posición de la lengua y los labios debido al temblor provocado por la EP. Para cada uno de los participantes se consideraron 13 características del tipo MFCC (MFCC0, MFCC1, ..., MFCC12).

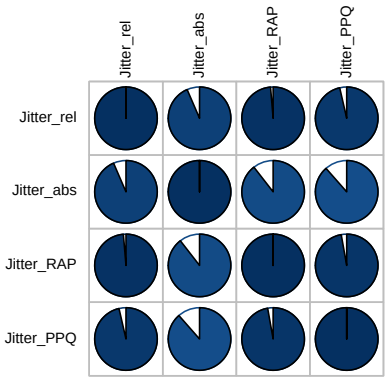
Se registraron otro tipo de características denominadas Delta, las cuales también son 13 (Delta0, Delta1, ..., Delta12), son derivadas del tiempo de las características MFCC, de modo que cada grabación tiene 26 mediciones de este tipo, 13 de la característica MFCC y 13 de la característica Delta.

3.4. Aplicación del modelo de MSV

Para poder desarrollar el modelo de MSV haremos uso del software `R` así como de la librería `e1071` contenida en éste. Siguiendo lo que hemos mencionado anteriormente, primero se dividirá la base de datos aleatoriamente en datos de prueba y de entrenamiento, siendo el 25 % (20 observaciones) y 75 % (60 observaciones), respectivamente.

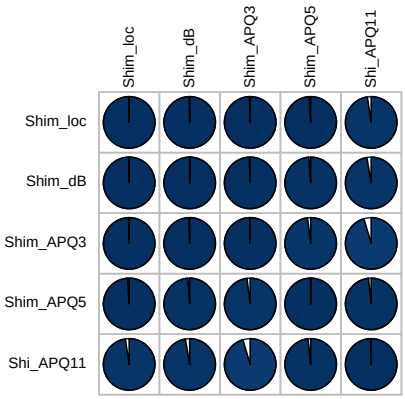
También nos interesa la correlación que exite entre cada grupo de variables, observemos que en las figuras 3.1, 3.2, 3.3, 3.4 y 3.5 se muestra una correlación muy alta y eso debemos considerarlo en nuestro modelo al momento de dar las características para éste, ya que no podremos poner características del mismo grupo. Las correlaciones entre las variables de distintos grupos son menores en comparación a las variables que pertenecen a los mismos grupos. Por ejemplo:

- Jitter_Absoluto vs Jitter_Relativo **mayor correlación**
- Jitter_Absoluto vs HNR_15 **menor correlación**



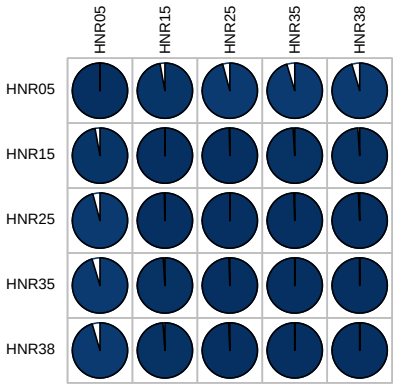
1	0.93	0.99	0.96
0.93	1	0.89	0.89
0.99	0.89	1	0.97
0.96	0.89	0.97	1

Figura 3.1: Parkinson, correlación entre el grupo de características Jitter.



1	1	1	0.99	0.98
1	1	0.99	0.99	0.97
1	0.99	1	0.98	0.95
0.99	0.99	0.98	1	0.98
0.98	0.97	0.95	0.98	1

Figura 3.2: Parkinson, correlación entre grupo de características Shimmer.



1	0.97	0.96	0.95	0.95
0.97	1	1	0.99	0.99
0.96	1	1	0.99	0.99
0.95	0.99	0.99	1	1
0.95	0.99	0.99	1	1

Figura 3.3: Parkinson, correlación entre grupo de características HNR.

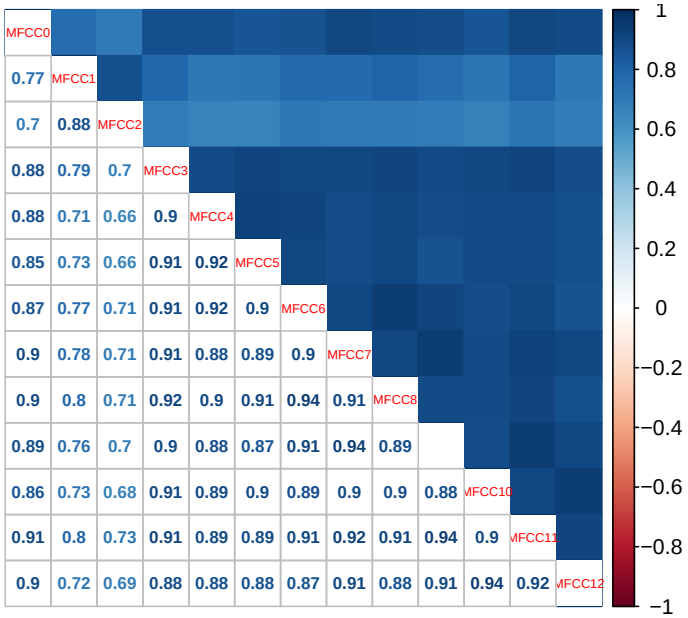


Figura 3.4: Parkinson, correlación entre el grupo de características MFCC.

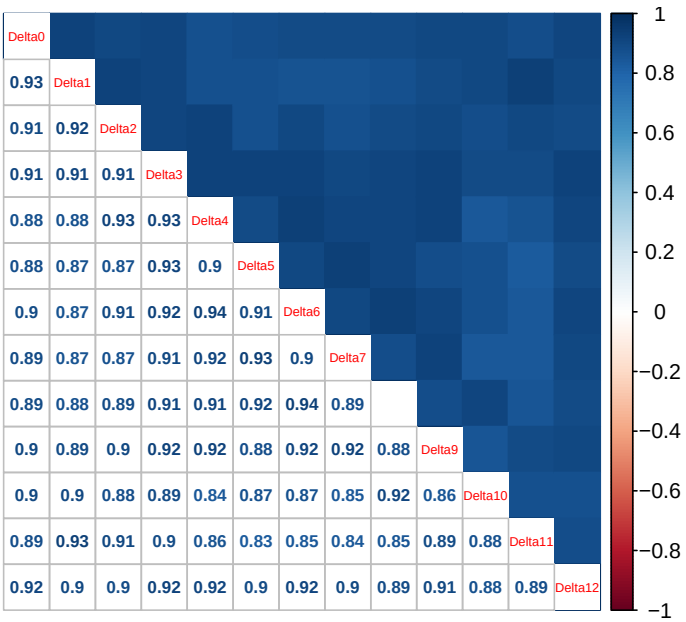


Figura 3.5: Parkinson, correlación entre el grupo de características Delta.

Ya que vimos la correlación y dividimos los datos aleatoriamente, nos resta aplicar el modelo de MSV. En este capítulo haremos el modelo con todas las características, sin embargo, también se hará una comparación con un modelo de sólo once características. Esto se hace para ver la eficacia y el error que muestra el modelo explicado con sólo parte de las variables ya que pueden existir problemas de multicolinealidad debido a las altas correlaciones que existen entre las características pertenecientes a los mismo grupos. El código de R se presenta en el Apéndice A.

3.4.1. Modelo con todas las variables

Empezaremos con el modelo explicado por todas las características. Como la variable respuesta de la base de datos es de tipo binaria, haremos el modelo de MSV empezando a usar un núcleo de tipo lineal con el parámetro $C = 2.2$, para poder determinar el mejor costo usamos la función `tune.svm` (ver Apéndice A) como se muestra a continuación:

```
Parameter tuning of svm:

- sampling method: 10-fold cross validation
```

```
- best parameters:
 cost
  2.2

- best performance: 0.1333333

- Detailed performance results:
    cost     error dispersion
1    0.2 0.1833333 0.12297746
2    0.4 0.1500000 0.12297746
3    0.6 0.1500000 0.12297746
11   2.2 0.1333333 0.10540926
12   2.4 0.1333333 0.10540926
14   2.8 0.1666667 0.11111111
15   3.0 0.1666667 0.11111111
18   3.6 0.1833333 0.12297746
19   3.8 0.1833333 0.12297746
20   4.0 0.2000000 0.13146844
21   4.2 0.2000000 0.13146844
43   8.6 0.1833333 0.09460770
44   8.8 0.1833333 0.09460770
53  10.6 0.1666667 0.07856742
54  10.8 0.1666667 0.07856742
59  11.8 0.1833333 0.09460770
60  12.0 0.1833333 0.09460770
```

Con este tipo de núcleo lineal, el modelo da como resultado 19 vectores de soporte, 8 de la clase 0 (sin EP) y 11 de la clase 1 (con EP), la figura 3.6 muestra el rendimiento del modelo. Un núcleo lineal clasifica correctamente a todos los datos de entrenamiento, sin embargo, al hacer la predicción con los datos de prueba son 4 los datos que predice erróneamente (ver tabla 3.6), es por ello que veremos el comportamiento que tiene este modelo con otros tipos de núcleo.

Conjunto de entrenamiento

clasificación \ y	0	1
0	30	0
1	0	30

Conjunto de prueba

predicción \ y	0	1
0	8	2
1	2	8

Tabla 3.6: Parkinson (44 variables), núcleo lineal. Tabla de clasificación y predicción.

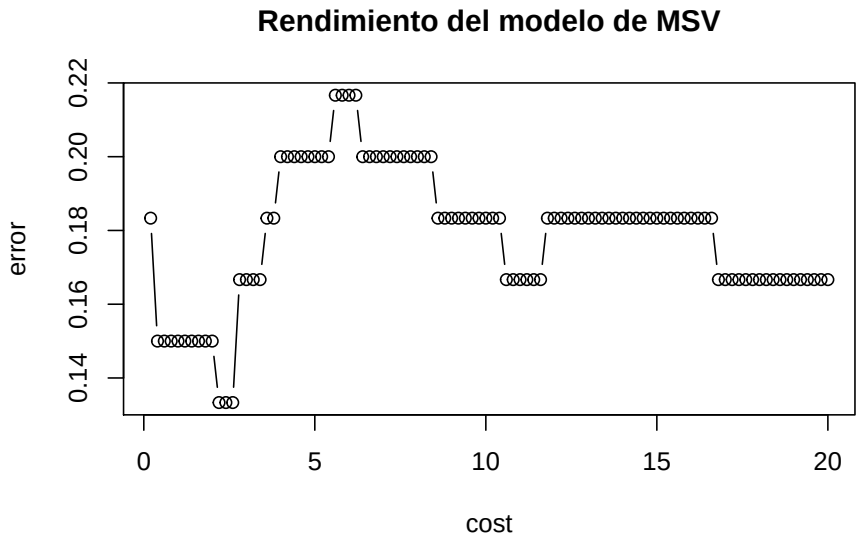


Figura 3.6: Parkinson (44 variables), núcleo lineal; rendimiento del modelo.

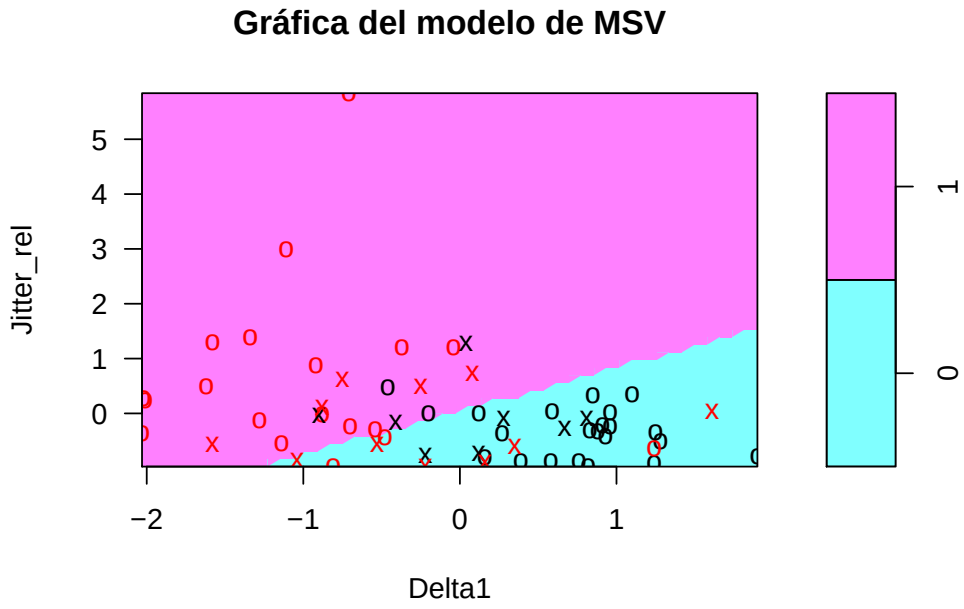


Figura 3.7: Parkinson (44 variables), núcleo lineal. Modelo de MSV con parámetro $C = 2.2$.

Probaremos el núcleo polinomial, con los parámetros $d = 2$ y $C = 0.2$ ya que

fueron los que arrojó la función `tune.svm`, sin embargo, este tipo de núcleo no funcionó para este modelo ya que tiene un margen de error muy alto y los datos no son bien clasificados (ver figura 3.9), como se mostrará en los resultados siguientes. La figura 3.8 muestra el rendimiento del modelo.

```
Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:
 degree cost
      2  0.2

- best performance: 0.1333333

- Detailed performance results:
    degree cost     error dispersion
1        2  0.2 0.1333333  0.1314684
2        3  0.2 0.1333333  0.1314684
3        4  0.2 0.1333333  0.1314684
4        5  0.2 0.1333333  0.1314684
5        6  0.2 0.1333333  0.1314684
6        7  0.2 0.1333333  0.1314684
7        8  0.2 0.1333333  0.1314684
8        9  0.2 0.1333333  0.1314684
9       10  0.2 0.1333333  0.1314684
10       2  0.4 0.1500000  0.1229775
11       3  0.4 0.1500000  0.1229775
12       4  0.4 0.1500000  0.1229775
13       5  0.4 0.1500000  0.1229775
```

En este modelo hay 58 vectores de soporte, 29 que pertenecen a la clase 0 y 29 pertenecientes a la clase 1, es decir sólo dos datos de entrenamiento no son vectores de soporte y desde aquí podemos deducir que no será un buen modelo ya que tendrá un mayor error de generalización. En la figura 3.9 podemos ver cómo clasifica la mayoría de los datos en la clase 0 y no se puede observar un plano de separación. Con el tipo de núcleo que utilizamos, a la clase 0 la clasifica correctamente, sin embargo de la clase 1 solamente clasifica 9 datos correctamente y por lo tanto estaría clasificando mal 21 datos. De acuerdo a la tabla 3.7 al hacer la predicción con los datos de prueba de la base de datos, el modelo estaría haciendo una no muy buena predicción.

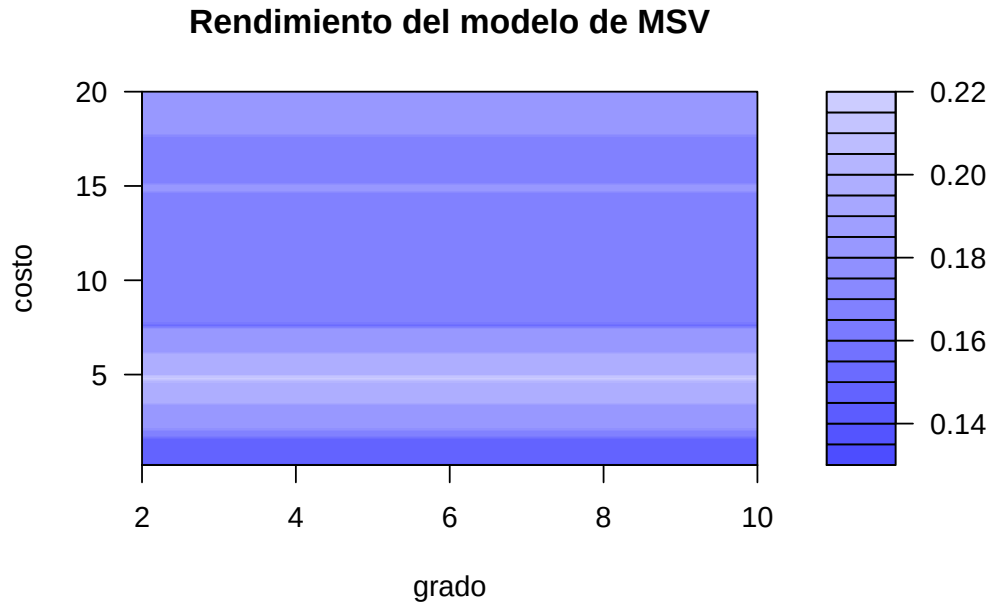


Figura 3.8: Parkinson (44 variables), núcleo polinomial; rendimiento del modelo.

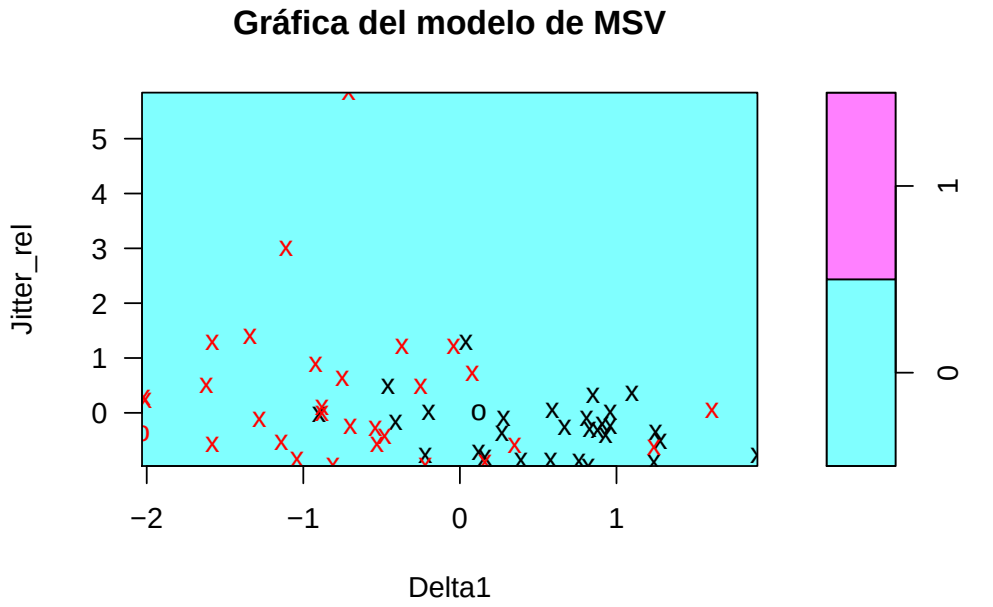


Figura 3.9: Parkinson (44 variables), núcleo polinomial. Modelo de MSV con parámetros $C = 0.2$ y $d = 2$.

Conjunto de entrenamiento

clasificación \ y	0	1
0	30	21
1	0	9

Conjunto de prueba

predicción \ y	0	1
0	9	9
1	1	1

Tabla 3.7: Parkinson (44 variables), núcleo polinomial. Tabla de clasificación y predicción.

A continuación se usará un núcleo radial en el modelo de MSV, con parámetros $\gamma = 0.01333333$ y $C = 2.8$ (ver figura 3.11), con este modelo hay 37 vectores de soporte, 21 que pertenecen a la clase 0 y 16 que pertenecen a la clase 1. Con el tipo de núcleo que utilizamos, 3 datos de la clase 0 son clasificados en la clase 1 y 2 datos que pertenencen a la clase 1 son mal clasificados en la clase 0 y la predicción hecha por este modelo solamente predice mal 5 datos en total (tabla 3.8). Este modelo podría ser una opción ya que sólo clasifica 5 datos erróneamente y los vectores de soporte apenas son más de la mitad de los datos de entrenamiento (ver figura 3.10).

Conjunto de entrenamiento

clasificación \ y	0	1
0	27	2
1	3	28

Conjunto de prueba

predicción \ y	0	1
0	8	3
1	2	7

Tabla 3.8: Parkinson (44 variables), núcleo radial. Tabla de clasificación y predicción.

Por último usaremos el modelo de MSV con un núcleo sigmoidal con los mismos parámetros que usamos en el núcleo radial, debido a que en los campos de la función *tune.svm* se expresan los mismos tanto para el núcleo radial como para el núcleo sigmoidal. Este modelo nos da como resultado 31 vectores de soporte, 16 que pertenecen a la clase 0 y 15 a la clase 1 (figura 3.12). Con este tipo de núcleo, 5 datos de la clase 0 son clasificados en la clase 1 y 3 datos que pertenencen a la clase 1 son mal clasificados en la clase 0 y al hacer la predicción solamente predice 2 datos erróneos (tabla 3.9).

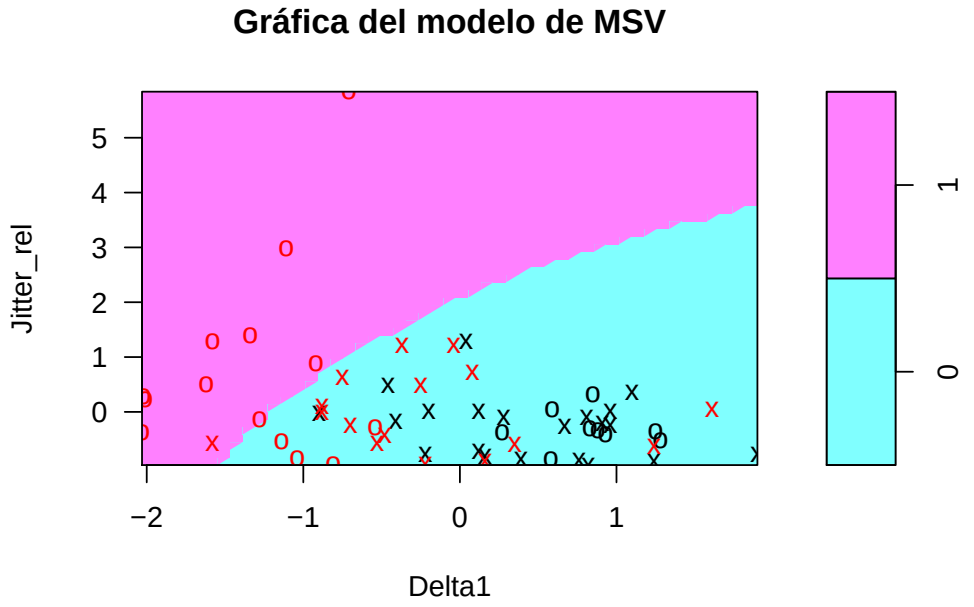


Figura 3.10: Parkinson (44 variables), núcleo radial. Modelo de MSV con parámetros $C = 2.8$ y $\alpha = 0.01333333$.

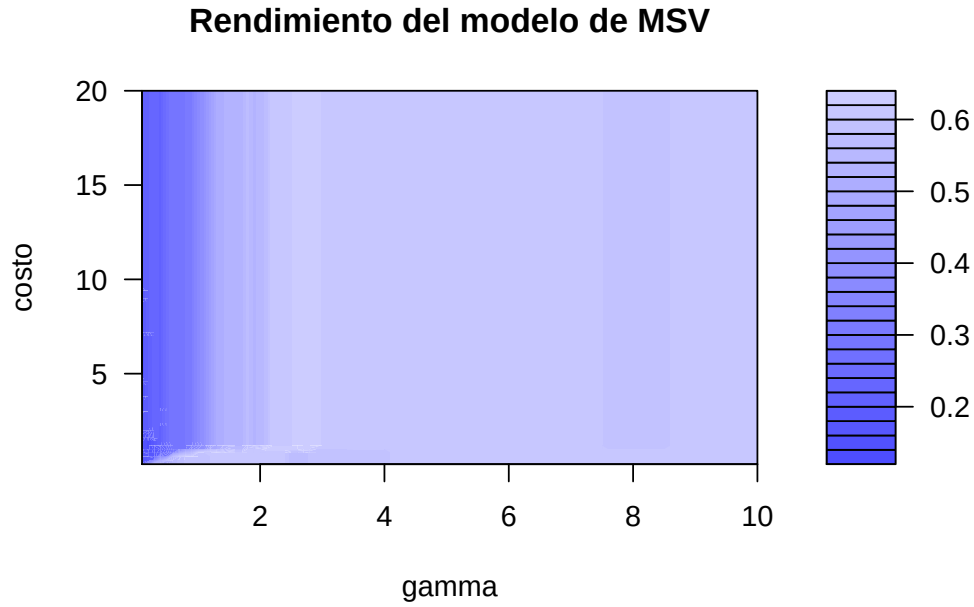


Figura 3.11: Parkinson (44 variables), núcleo radial; rendimiento del modelo.

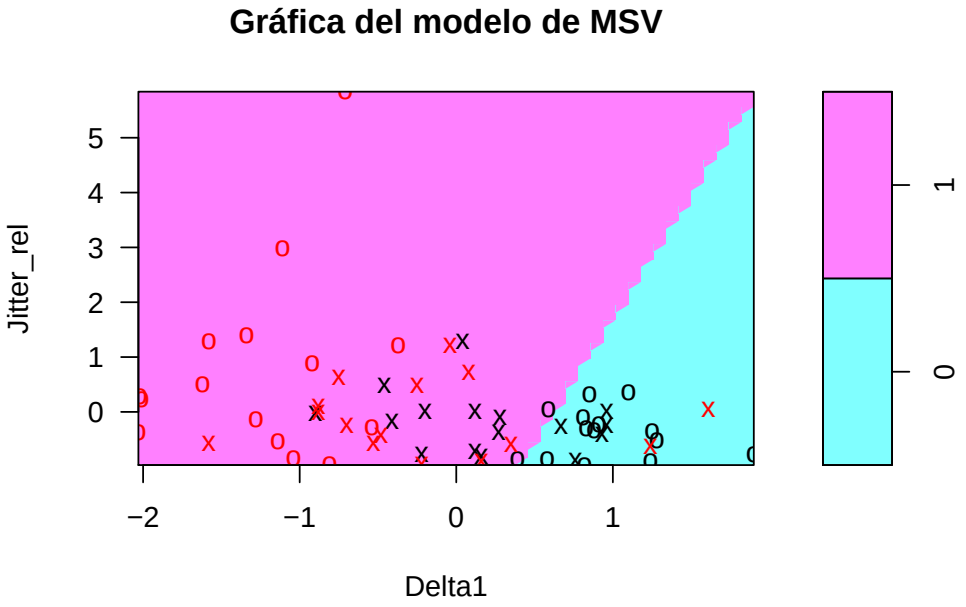


Figura 3.12: Parkinson (44 variables), núcleo sigmoidal. Modelo de MSV con parámetros $C = 2.8$ y $\alpha = 0.01333333$.

Notamos que aunque usamos los mismos parámetros en este tipo de núcleo sigmoidal aumentan 3 datos que están mal clasificados, es decir en este modelo en lugar de 5 son 8 los datos que no son correctamente clasificados, sin embargo, hay un número menor de vectores de soporte y en comparación con los modelos anteriores este modelo con un núcleo sigmoidal es el que hace la mejor predicción.

Conjunto de entrenamiento

clasificación \ y	0	1
0	25	3
1	5	27

Conjunto de prueba

predicción \ y	0	1
0	9	1
1	1	9

Tabla 3.9: Parkinson (44 variables), núcleo sigmoidal. Tabla de clasificación y predicción.

Después de este análisis se deduce que el modelo de MSV para esta base de datos es el que se realiza con el núcleo sigmoidal ya que se podría decir

discrimina correctamente a las dos clases y aparte es el modelo que mejor predice a los datos de prueba de la base de datos. Es por ello que este modelo es el óptimo para poder clasificar y predecir a los participantes con EP y sin EP.

3.4.2. Modelo con 11 variables

Ahora haremos una comparación con el mismo modelo a diferencia de que éste sólo tendrá 11 características, éstas son: Jitter_rel, Shim_loc, HNR05, RPDE, DFA, PPE, GNE, MFCC2, MFCC5, Delta5 y Delta11. Para poder escoger estas variables se eligió una variable de cada grupo de Jitter, Shimmer, HNR, se eligieron todas las mediciones no lineales, y para el grupo formado por las características MFCC y Delta se sacó la correlación una a una dentro de cada grupo de características y se tomó la que tuviera menos correlación entre cada grupo de características. Para la selección de variables se podrían usar métodos de selección de variables (como forward o backward), reducción de la dimensión (como componentes principales) o métodos de ponderación de variables (como LASSO), como los que se mencionan en (Hastie et al., 2001).

Esto para poder ver qué tan eficiente es el modelo con tan sólo este número de características ya que existe mucha correlación entre cada grupo, se espera que el resultado al aplicar el modelo de MSV sea muy similar.

El modelo de MSV inicial será usando un núcleo lineal con $C = 0.6$ (se escogió este parámetro con la función `tune.svm`, se observa su comportamiento en la figura 3.13.), este modelo nos arroja 26 vectores de soporte, 13 que pertencen a la clase 0 y 13 pertenecientes a la clase 1 (ver figura 3.14). Con el tipo de núcleo que utilizamos, dos de los datos de la clase 0 son clasificados en la clase 1 y 3 datos que pertenencen a la clase 1 son mal clasificados en la clase 0, por lo tanto se clasifican erróneamente 5 datos de entrenamiento a comparación del modelo con todas las características que sí clasificó correctamente a todos los datos, sin embargo, en la tabla 3.10 se observa que la predicción en este modelo es mucho mejor ya que fue la predicción que conseguimos en el modelo anterior pero con un núcleo sigmoidal, entonces, aunque haya tenido un error más grande de clasificación, su predicción fue mejor.

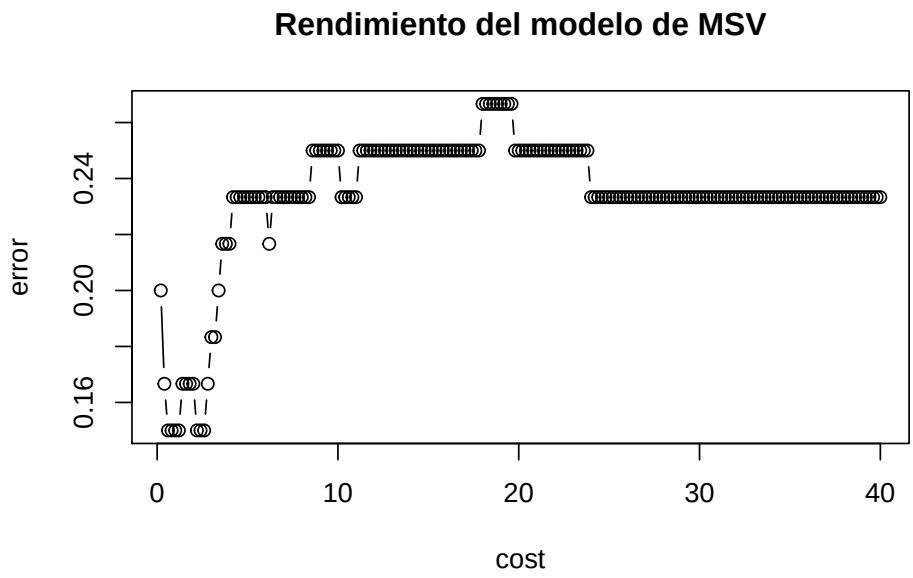


Figura 3.13: Parkinson (11 variables), núcleo lineal; rendimiento del modelo.

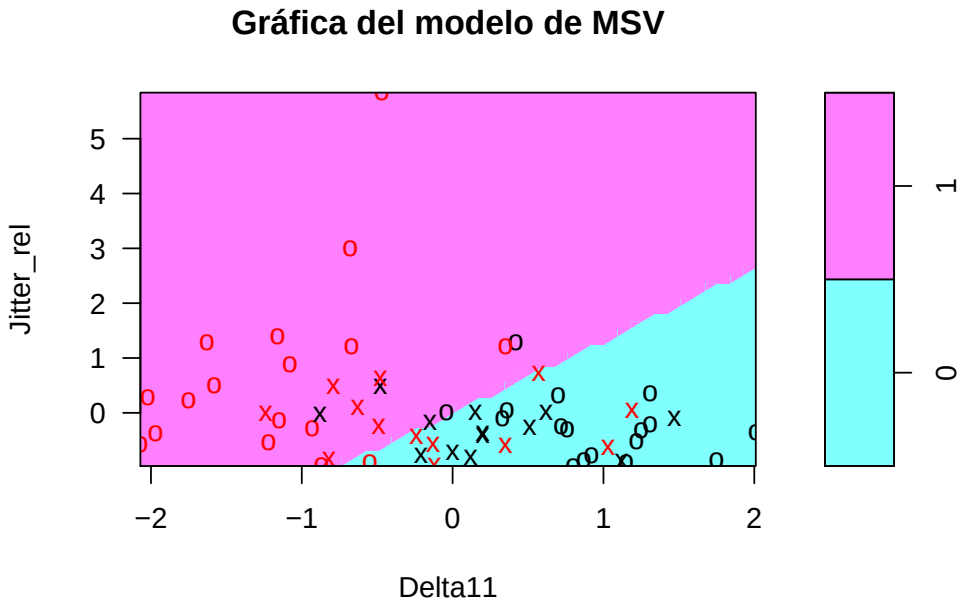


Figura 3.14: Parkinson (11 variables), núcleo lineal. Modelo MSV con parámetro $C = 0.6$.

Conjunto de entrenamiento

clasificación \ y	0	1
0	28	3
1	2	27

Conjunto de prueba

predicción \ y	0	1
0	9	1
1	1	9

Tabla 3.10: Parkinson (11 variables), núcleo lineal. Tabla de clasificación y predicción.

Con un modelo de MSV usando un núcleo polinomial con parámetros $d = 2$ y $C = 1.4$ (escogidos por la función `tune.svm`, ver figura 3.15). 10 de los datos de entrenamiento son mal clasificados (ver figura 3.16), este modelo es similar al que contiene todas las características ya que no es muy eficiente al clasificar los datos, sin embargo, hay menos datos erróneamente clasificados que en el otro. En la figura 3.15 se muestra el rendimiento del modelo. En este modelo hay 50 vectores de soporte, 27 que pertenecen a la clase 0 y 23 pertenecientes a la clase 1. Con el tipo de núcleo que utilizamos, un sólo dato de la clase 0 está mal clasificado en la clase 1 y 9 datos que pertenencen a la clase 1 son mal clasificados en la clase 0, al hacer la predicción tampoco resulta ser muy eficiente ya que, al igual que en el modelo con todas las variables, no es una buena predicción (ver tabla 3.11). Quizá lo que esté sucediendo es que el núcleo polinomial se esté sobreajustando a los datos de entrenamiento. Lo que se espera es que este algoritmo alcance un estado en el que sea capaz de predecir el resultado en otros casos a partir de lo aprendido con los datos de entrenamiento para poder resolver situaciones distintas a las que se tienen durante el entrenamiento. En este caso el algoritmo queda ajustado a unas características muy específicas de los datos de entrenamiento y cuando responde a muestras de entrenamiento nuevas va empeorando.

Conjunto de entrenamiento

clasificación \ y	0	1
0	29	9
1	1	21

Conjunto de prueba

predicción \ y	0	1
0	8	7
1	2	3

Tabla 3.11: Parkinson (11 variables), núcleo polinomial. Tabla de clasificación y predicción.

Probando el núcleo radial con parámetros $C = 0.8$ y $\gamma = 0.08$ (figura 3.17)

da como resultado 43 vectores de soporte (ver figura 3.18), 22 que son de la clase 0 y 21 que son de la clase 1. Este modelo es parecido al modelo con todas las características ya que el número de vectores de soporte al igual que en el otro modelo son más de la mitad de los datos de entrenamiento y eso podría afectar con la eficiencia del modelo. Con este modelo sólo se clasifican mal 6 datos, 4 datos de la clase 0 están clasificados en la clase 1 y 2 datos que pertenencen a la clase 1 son mal clasificados en la clase 0, notamos que hay una buena predicción al sólo predecir erróneamente 3 datos (ver tabla 3.12), sin embargo, la mejor predicción sigue siendo, hasta este momento, la del modelo inicial con un núcleo lineal.

Conjunto de entrenamiento

clasificación \ y	0	1
0	26	2
1	4	28

Conjunto de prueba

predicción \ y	0	1
0	9	2
1	1	8

Tabla 3.12: Parkinson (11 variables), núcleo radial. Tabla de clasificación y predicción.

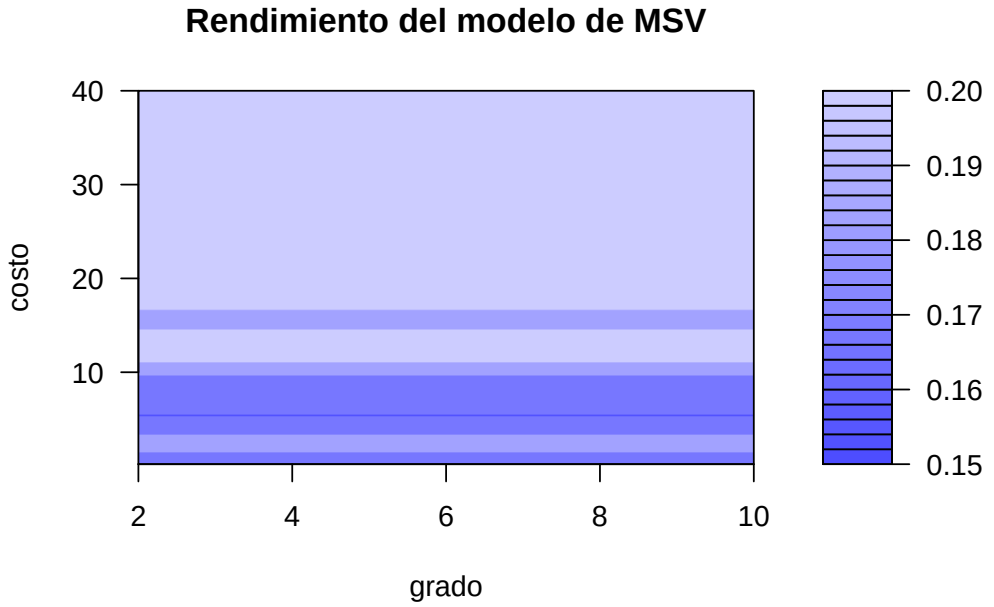


Figura 3.15: Parkinson (11 variables), núcleo polinomial; rendimiento del modelo.

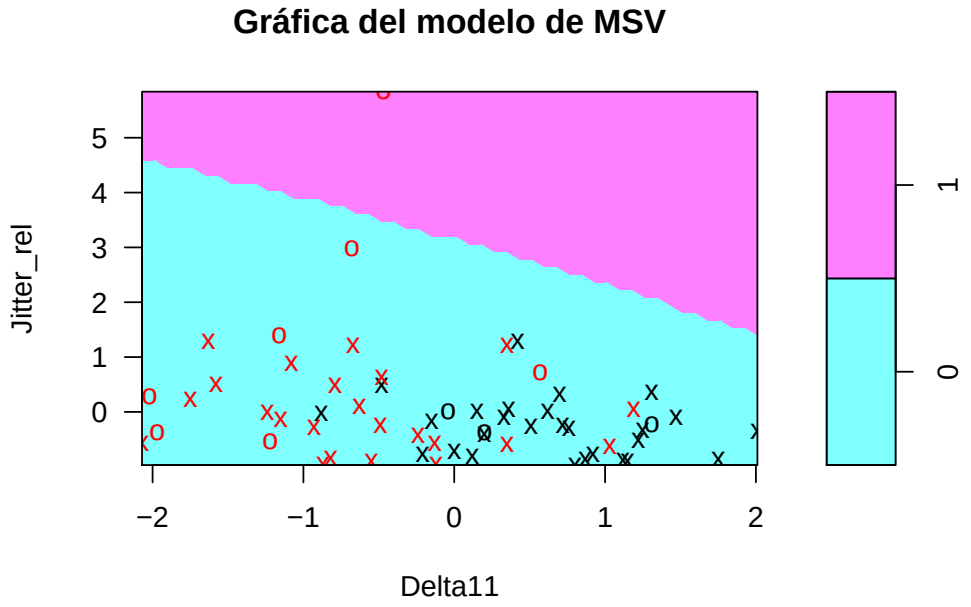


Figura 3.16: Parkinson (11 variables), núcleo polinomial. Modelo de MSV con parámetros $C = 5.4$ y $d = 2$.

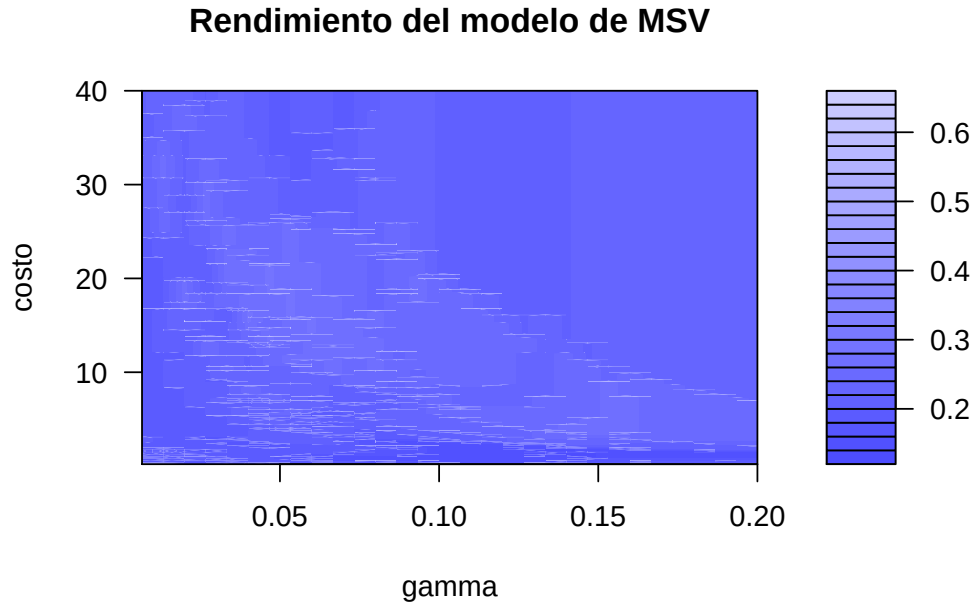


Figura 3.17: Parkinson (11 variables), núcleo radial; rendimiento del modelo.

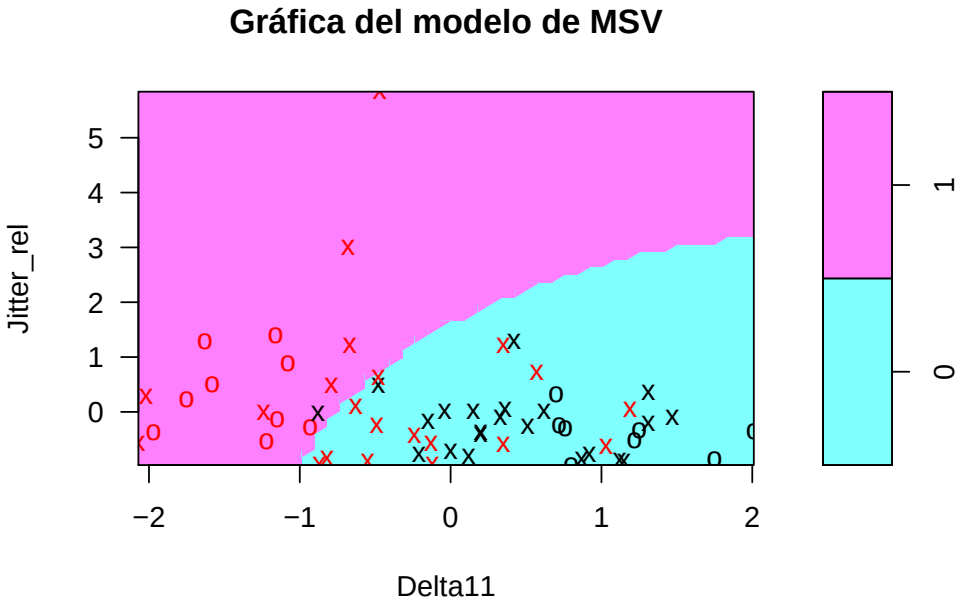


Figura 3.18: Parkinson (11 variables), núcleo radial. Modelo de MSV con parámetros $C = 0.8$ y $\gamma = 0.08$.

Probemos por último con un modelo usando un núcleo sigmoidal con parámetros $C = 0.8$ y $\gamma = 0.08$, este modelo nos da como resultado 33 vectores de soporte, 16 que pertenecen a la clase 0 y 17 a la clase 1 (ver figura 3.19). Aquí solamente son mal clasificados 11 datos, 4 datos de la clase 0 están clasificados en la clase 1 y 7 datos que pertenencen a la clase 1 son mal clasificados en la clase 0. La predicción que hace este modelo es la misma que la del modelo utilizando un núcleo radial (ver tabla 3.13), entonces podemos decir que en el modelo donde están todas las características la predicción fue mejor y en este caso con sólo 11 características la mejor predicción es la del modelo utilizando un núcleo lineal.

Conjunto de entrenamiento

clasificación \ y	0	1
0	25	7
1	4	27

Conjunto de prueba

predicción \ y	0	1
0	9	2
1	1	8

Tabla 3.13: Parkinson (11 variables), núcleo sigmoidal. Tabla de clasificación y predicción.

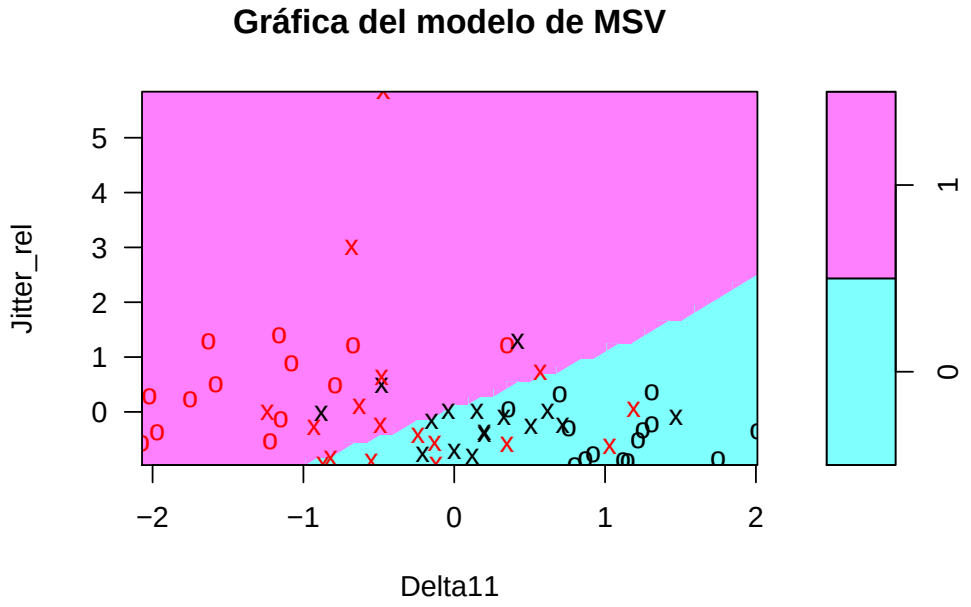


Figura 3.19: Parkinson (11 variables), núcleo sigmoidal. Modelo de MSV con parámetros $C = 0.8$ y $\alpha = 0.08$.

3.4.3. Modelo con 11 variables y 3 conjuntos de datos

Para poder dar un resultado más certero, se procedió a hacer un tercer conjunto de datos, es decir, dividir nuestra base de datos en datos de entrenamiento, datos de validación y datos de prueba.

La función de estos conjuntos de datos es la siguiente:

- Muestra de entrenamiento: son los datos con los que se entrenan (estiman o ajustan) los modelos.
- Muestra de validación: selecciona el mejor de los modelos entrenados.
- Muestra de prueba: calcula el error real cometido con el modelo seleccionado.

Dicho esto en esta sección hemos hecho la estratificación de los datos de la siguiente manera: entrenamiento 50 % de los datos (20 personas sin EP y 20 con EP), validación 25 % de los datos (10 personas con EP y 10 sin EP) y

prueba 25 % de los datos (10 personas con EP y 10 sin EP). Entonces con los datos de entrenamiento se probaron todos los núcleos: lineal, polinomial, radial y sigmoidal.

Para este proceso se simularon 100 muestras aleatorias con el conjunto de datos del modelo con 11 características. Ajustamos el modelo con los datos de entrenamiento para los distintos núcleos (lineal, polinomial, radial y sigmoidal), el resultado que se obtuvo al sacar el promedio de los parámetros para este proceso fue el siguiente:

Núcleo	Parámetros
Lineal	$C = 1.142$
Polinomial	$C = 0.8$, $d = 2$
Radial	$C = 1.972$, $\gamma = 0.0298$
Sigmoidal	$C = 2.082$, $\gamma = 0.0258$

Tabla 3.14: Promedio de los parámetros de los distintos núcleos de las 100 muestras aleatorias.

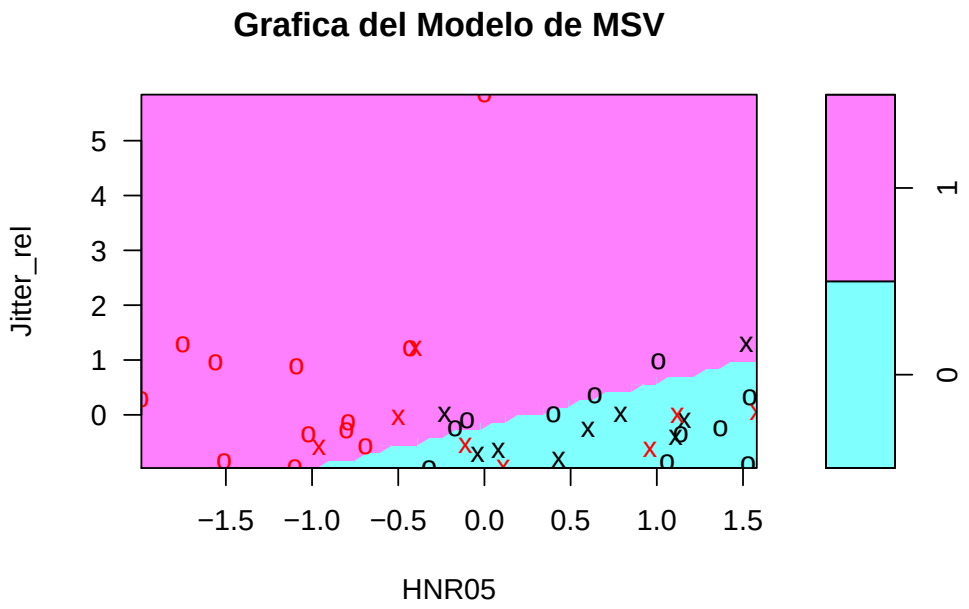


Figura 3.20: Parkinson (11 variables con 3 conjuntos de datos), núcleo lineal. Modelo de MSV con parámetro $C = 1.142$.

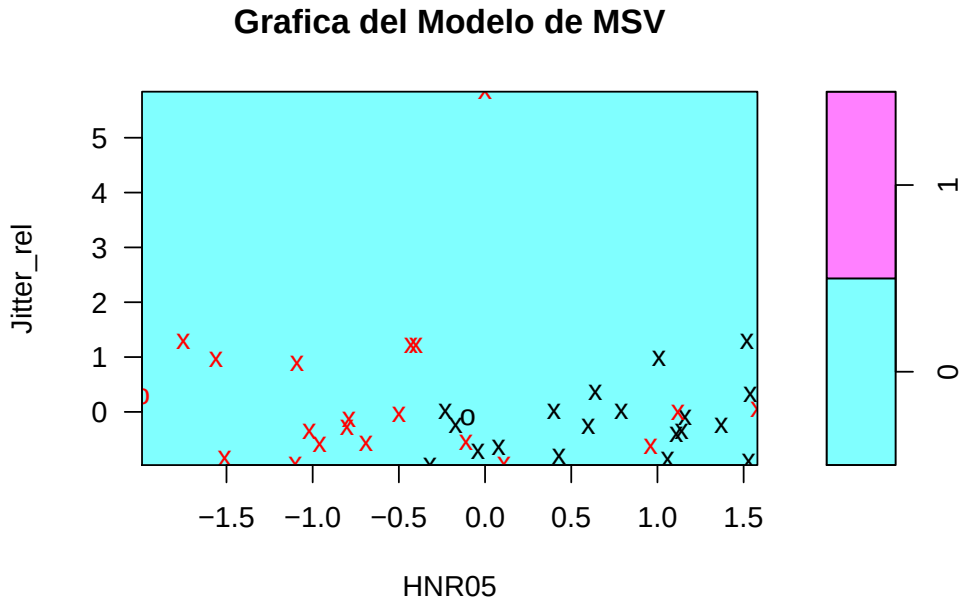


Figura 3.21: Parkinson (11 variables con 3 conjuntos de datos), núcleo polinomial. Modelo de MSV con parámetros $C = 0.8$ y $d = 2$.

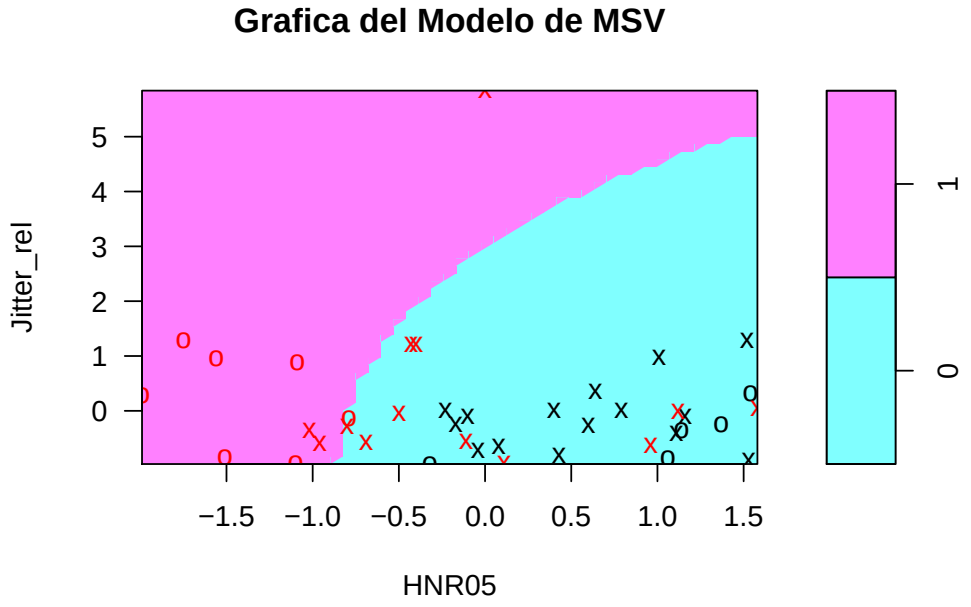


Figura 3.22: Parkinson (11 variables con 3 conjuntos de datos), núcleo radial. Modelo de MSV con parámetros $C = 1.972$ y $\gamma = 0.0298$.

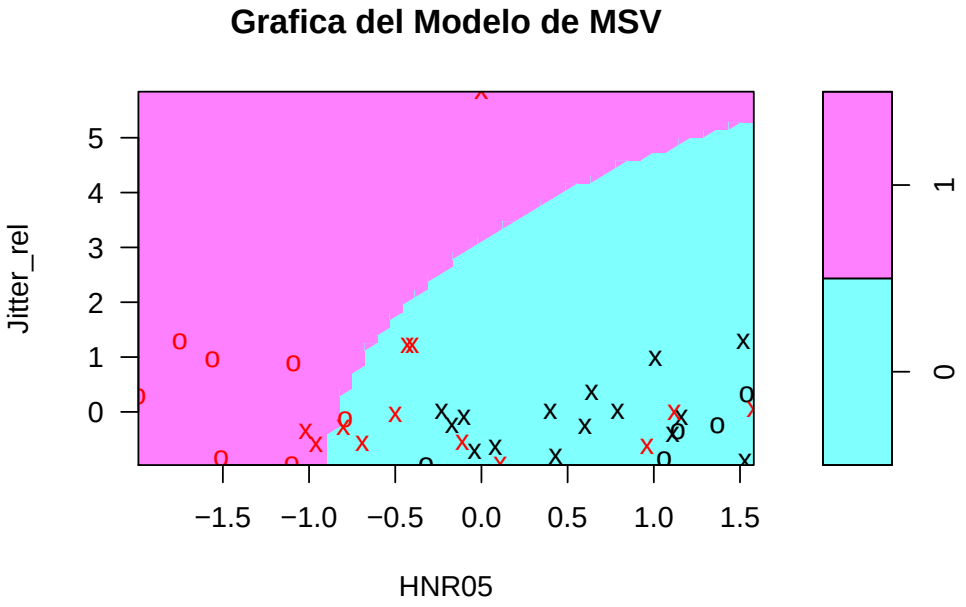


Figura 3.23: Parkinson (11 variables con 3 conjuntos de datos), núcleo sigmoidal. Modelo de MSV con parámetros $C = 2.082$ y $\gamma = 0.0258$.

Después de esto lo que hicimos fue seleccionar el mejor modelo de los que entrenamos con los datos anteriores, para este proceso utilizamos el conjunto de datos de validación, con lo cual obtuvimos los siguientes resultados:

Conjunto de validación

validación \ y	0	1
0	8	2
1	2	8

Tabla 3.15: Parkinson (11 variables, 3 conjuntos de datos), núcleo lineal. Tabla de resultados con el conjunto de validación.

Conjunto de validación

y / validación	0	1
0	9	7
1	2	3

Tabla 3.16: Parkinson (11 variables, 3 conjuntos de datos), núcleo polinomial. Tabla de resultados con el conjunto de validación.

Conjunto de validación

y / validación	0	1
0	7	1
1	3	9

Tabla 3.17: Parkinson (11 variables, 3 conjuntos de datos), núcleo radial. Tabla de resultados con el conjunto de validación.

Conjunto de validación

y / validación	0	1
0	7	1
1	3	9

Tabla 3.18: Parkinson (11 variables, 3 conjuntos de datos), núcleo sigmoidal. Tabla de resultados con el conjunto de validación.

Después de estos resultados, proseguimos a seleccionar el mejor modelo para así poder aplicarlo al conjunto de datos de prueba (20 datos: 10 con EP y 10 sin EP) y poder ver el error real cometido por el modelo. El modelo más óptimo que eligimos fue el modelo usando el núcleo lineal (ver tabla 3.15) ya que nos arroja el mismo resultado que los núcleos sigmoidal (ver tabla 3.17) y radial (ver tabla 3.18), sin embargo, el núcleo lineal es un modelo menos complejo y es por ello que lo hemos elegido como el modelo óptimo y es con el que vamos a usar el conjunto de prueba. Al usar el conjunto de datos de prueba en el núcleo lineal, obtuvimos que un solo dato fue erróneamente

clasificado en la clase 1 y dos datos fueron mal clasificados en la clase 0 (ver tabla 3.19).

Conjunto de prueba

prueba \ y	0	1
0	8	1
1	2	9

Tabla 3.19: Parkinson (11 variables, 3 conjuntos de datos), núcleo lineal. Tabla de resultados con el conjunto de prueba.

3.4.4. Resultados, validación cruzada

En la tabla 3.20 se muestran los resultados obtenidos en las dos secciones anteriores con una sola muestra. La validación cruzada es un método que sirve para evaluar el rendimiento de un modelo. En este caso este procedimiento se realiza cien veces de manera automática y lo que se reporta son las medias de cada una de las características, es decir, el número de vectores de soporte, los parámetros, los datos mal clasificados y los mal predecidos (ver tabla 3.21).

	núcleo	Parámetros	Vectores de soporte	Datos mal clasificados (conjunto de entrena-miento)	Datos mal predecidos (conjunto de prueba)
Modelo 44 variables	Lineal	$C = 2.2$	19	0	4
	Polinomial	$C = 0.2$ $d = 2$	58	9	10
	Radial	$\gamma =$ 0.013333333 $C = 2.8$	37	5	5
	Sigmoidal	$\gamma =$ 0.013333333 $C = 2.8$	31	8	2
Modelo 11 variables	Lineal	$C = 0.6$	26	5	2
	Polinomial	$C = 1.4$ $d = 2$	50	10	9
	Radial	$C = 0.8$ $\gamma = 0.08$	45	8	3
	Sigmoidal	$C = 0.8$ $\gamma = 0.08$	36	11	3

Tabla 3.20: Parkinson, tabla de resultados finales con muestra única.

	núcleo	Parámetros	Vectores de soporte	Datos mal clasificados (conjunto de entrena-miento)	Datos mal predecidos (conjunto de prueba)
Modelo 44 variables	Lineal	$C = 1.118$	26 (3.02)	1.9 (1.16)	4.76 (1.62)
	Polinomial	$C = 0.2$ $d = 2$	57.21 (2.68)	22.95 (1.3)	9.27(1.74)
	Radial	$\gamma = 0.0156$ $C = 1.0496$	38.38 (2.54)	7.09(1.34)	3 (1.41)
	Sigmoidal	$\gamma = 0.0156$ $C = 1.0496$	32 (2.2)	7.55	2 (1.1)
Modelo 11 variables	Lineal	$C = 0.664$	25.06 (2.61)	5.7 (1.5)	3.5 (1.4)
	Polinomial	$C = 1.4$ $d = 2$	55.91 (3.04)	19.11 (1.82)	8.76 (1.65)
	Radial	$C = 2.894$ $\gamma = 0.0242$	33.25 (2.4)	6.5 (1.4)	3.57 (1.5)
	Sigmoidal	$C = 2.894$ $\gamma = 0.0242$	33.8 (2.1)	7.35 (1.4)	3.1 (1.4)

Tabla 3.21: Parkinson, tabla de resultados finales. Media (desviación estándar) de un total de 100 conjuntos de datos.

	Núcleo	Parámetros	Vectores de soporte	Datos mal clasificados (conjunto de validación)	Datos mal predecidos (conjunto de prueba)
Modelo 11 variables	Lineal	$C = 1.142$	18.02 (3.24)	3.8 (1.58)	3 (1.73)
	Polinomial	$C = 0.8$ $d = 2$	38.08 (1.15)	8.55 (1.58)	
	Radial	$C = 1.972$ $\gamma = 0.0298$	25.73 (2.57)	3.52 (1.62)	
	Sigmoidal	$C = 2.082$ $\gamma = 0.0258$	25.97 (2.28)	3.54 (1.47)	

Tabla 3.22: Parkinson, tabla de resultados finales con 3 conjuntos de datos. Media (desviación estándar) de un total de 100 conjuntos de datos.

Capítulo 4

Conclusiones

Al hacer un análisis de la base de datos y aplicar el modelo de máquinas de soporte vectorial, podemos ver que todo lo dicho anteriormente es efectivamente cierto, una máquina de aprendizaje observa n pares de datos (x_i, y_i), en este caso cada x_i es un vector de variables características de la base de datos, y_i es la variable respuesta la regla de decisión es la siguiente:

$y = 1$ si los datos x_i pertenecen a los que tienen la EP.

$y = 0$ si los datos x_i pertenecen a los que no tienen la EP.

El tipo de clasificación que se usó fue una clasificación binaria con datos no linealmente separables, se concluye que el modelo óptimo para esta base de datos fue usar un núcleo lineal para la base de datos con 11 de las 44 características, esto debido a que con el núcleo lineal es un modelo menos complejo a comparación de los otros núcleos, es decir el polinomial, radial y sigmoidal.

Para poder llegar a este resultado hicimos uso de dos distintos métodos:

1. Consistió en dividir nuestra base de datos en dos conjuntos de datos: datos de entrenamiento y datos de prueba.

2. Consistió en dividir nuestra base de datos en tres conjuntos de datos: datos de entrenamiento, datos de validación y datos de prueba.

Tanto en el primer método como en el segundo hemos obtenido el mismo resultado, en los dos casos el núcleo lineal ha sido el mejor modelo o el modelo óptimo. En el caso 1 el núcleo lineal con los datos de prueba (40 sujetos: 20 con EP y 20 sin EP) tan solo discriminó erróneamente 4 datos y en el segundo caso con los datos de prueba (20 sujetos: 10 con EP y 10 sin EP) el núcleo lineal clasificó mal 3 datos.

Las ventajas de una máquina de soporte vectorial son las siguientes:

1. Maximización de la capacidad de generalización. Al entrenar un clasificador, especialmente cuando los datos de entrenamiento son escasos y linealmente separables, la capacidad de generalización se deteriora considerablemente, pero debido a que una máquina de soporte vectorial está entrenada para maximizar el margen, la capacidad de generalización no se deteriora mucho, incluso bajo tal condición.

2. Sin mínimos locales. Debido a que una MSV se formula como un problema de programación cuadrática, existe una solución óptima global.

3. Amplia gama de aplicaciones. En las MSV, la capacidad de generalización se controla cambiando un núcleo, su parámetro y el parámetro del margen.

4. Robustez a valores atípicos. En las MSV, el parámetro de margen C controla el error de clasificación errónea. Si un valor grande se establece en C, la clasificación errónea se suprime, y si se establece un valor pequeño, los datos de capacitación que están alejados de los datos recopilados pueden clasificarse incorrectamente. Por lo tanto, al establecer correctamente un valor en C, podemos suprimir valores atípicos.

Las desventajas de la MSV son las siguientes:

1. Extensión a problemas multiclase. Las MSV usan funciones de decisión directa, por lo tanto, una extensión a los problemas multiclase no es directa y hay varias formulaciones.

2. Tiempo de entrenamiento largo. Debido a que el entrenamiento de una MSV se hace resolviendo el problema dual asociado, el número de variables es igual al número de datos de entrenamiento, por lo tanto, para una gran cantidad de datos de entrenamiento resolver el problema dual se vuelve difícil tanto para el tamaño de la memoria como para el tiempo de entrenamiento.

3. Selección de parámetros. Al entrenar una MSV necesitamos seleccionar un núcleo apropiado, sus parámetros y establecer el valor en el parámetro de margen C. Seleccionar los parámetros óptimos para un problema dado se llama selección de modelo. La selección del modelo se realiza mediante la estimación de la capacidad de generalización a través del entrenamiento repetido de MSV. Pero debido a que esto consume mucho tiempo, se han propuesto varios índices para estimar la capacidad de generalización.

Algunas variantes que también podemos estudiar con las máquinas de soporte vectorial son:

- Máquinas de soporte vectorial para mínimos cuadrados.
- Máquinas de soporte vectorial de programación lineal.
- Máquinas de soporte vectorial bayesianas.

Concluimos entonces que se cumplió el objetivo de este modelo de MSV, poder clasificar correctamente las dos clases, es decir los que tienen EP y los que no tienen EP. Comprobamos que este modelo es más eficiente y puede discriminar correctamente a 2 o más clases. Así también concluimos que el modelo óptimo es eligiendo para esta base de datos un núcleo lineal con el modelo de 11 variables ya que los datos mal predecidos en este modelo son menores que en los otros modelos.

Comparado con otros modelos como con el modelo de regresión logística con el cuál también se estudió esta base de datos para el mismo propósito (para más detalle ver (Gutiérrez, 2017)), el modelo de MSV tiene una mayor capacidad de generalización, ya que obtuvimos que con el modelo de máquinas de soporte vectorial se obtienen mejores resultados que usando el modelo de regresión logística ya que con este modelo se predicieron mal 4 datos a comparación del nuestro que se predicieron mal 3 datos (ver tabla 3.21 y 3.22).

Apéndice A

Código de R

```
#MODELO USANDO TODAS LAS VARIABLES
#Se cargan las librerias
library(lattice)
library(ggplot2)
library(caret)
library(e1071)
#Se cargan los datos
datos<-read.csv("C:/Users/ARTURO/Desktop/parkinson2.csv"
   , header= TRUE)
attach(datos)
names(datos)
#Se crean los grupos de variables para obtener la
   correlacion entre ellos
Jitter=cbind(Jitter_rel,Jitter_abs,Jitter_RAP,
Jitter_PPQ)
Shimmer=cbind(Shim_loc,Shim_dB,Shim_APQ3,Shim_APQ5,
Shi_APQ11)
HNR=cbind(HNR05,HNR15,HNR25,HNR35,HNR38)
MFCC=cbind(MFCC0,MFCC1,MFCC2,MFCC3,MFCC4,MFCC5,MFCC6,
   MFCC7,MFCC8,MFCC9,MFCC10,MFCC11,MFCC12)
Delta=cbind(Delta0,Delta1,Delta2,Delta3,Delta4,Delta5,
   Delta6,Delta7,Delta8,Delta9,Delta10,Delta11,Delta12)
#Se obtienen sus matrices de covarianzas
covJitter=cov(Jitter)
covShimmer=cov(Shimmer)
covHNR=cov(HNR)
covMFCC=cov(MFCC)
```

```
covDelta=cov(Delta)
#se obtienen las correlaciones
corJitter=cor(Jitter)
corShimmer=cor(Shimmer)
corHNR=cor(HNR)
corMFCC=cor(MFCC)
corDelta=cor(Delta)
#Se grafican los resultados anteriores
library(corrplot)
#JITTER
corrplot(corJitter,method = "pie",tl.col = "black",tl.
   cex = 0.6,cl.pos = "n")
corrplot(corJitter,method = "number",tl.col = "black",tl
   .cex = 0.6,cl.pos = "1")
#SHIMMER
corrplot(corShimmer,method = "pie",tl.col = "black",tl.
   cex = 0.6,cl.pos = "n")
corrplot(corShimmer,method = "number",tl.col = "black",
   tl.cex = 0.6,cl.pos = "1")
#HNR
corrplot(corHNR,method = "pie",tl.col = "black",tl.cex =
    0.6,cl.pos = "n")
corrplot(corHNR,method = "number",tl.col = "black",tl.
   cex = 0.6,cl.pos = "1")
#MFCC Y DELTA
corrplot.mixed(corMFCC,lower="number",upper = "color",
   tl.cex=0.5,number.cex=0.7)
corrplot.mixed(corDelta,lower = "number",upper="color",
   tl.cex=0.5,number.cex=0.7)
#Se dividen los datos aleatoriamente en datos de
   entrenamiento y de prueba -> 75%
#de los datos en datos de entrenamiento y 25% de los
   datos en datos de prueba
set.seed(998)
inTraining <- createDataPartition(datos$Estado, p = .75,
    list = FALSE)
training <- datos[ inTraining,]
testing  <- datos[-inTraining,]
################################################
#Hacemos el modelo de maquinas de soporte vectorial
#Se prueban distintos n\'ucleo

#n\'ucleo lineal
```

```
#Funcion tune.svm para poder escojer los parametros
   optimos
obj2 <- tune.svm(factor(Estado)~., data = training,cost=
   c(1:100)/5)
summary(obj2)
plot(obj2, main="Rendimiento del modelo de MSV")
#Modelo de MSV
model2 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "linear",
   cost = 2.2)
summary(model2)
#Graficar el modelo de MSV
plot(model2, training, Jitter_rel~ Delta1)
#Clasificacion
clas2<-predict(model2,training)
table(clas2,training$Estado) #clasifica todas bien
#prediccion
pred2<-predict(model2,testing)
table(pred2,testing$Estado)


#n\'ucleo polinomial
#Funcion tune.svm para poder escojer los parametros
   optimos
obj1 <- tune.svm(factor(Estado)~., data = training, cost
   =c(1:100)/5, degree = c(2:10))
summary(obj1)
plot(obj1,main="Rendimiento del modelo de MSV", xlab="
   grado", ylab="costo")
#Modelo de MSV
model1 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "polynomial",
    cost = 0.2 ,degree=2,coef.0=0)
summary(model1)
#Graficar el modelo de MSV
plot(model1, training, Jitter_rel~ Delta1)
#Clasificacion
clas<-predict(model1,training)
table(clas,training$Estado)
#Prediccion
pred<-predict(model1,testing)
table(pred,testing$Estado)
```

```
#n\'ucleo radial
#Funcion tune.svm para poder escojer los parametros
   optimos
obj3 <- tune.svm(factor(Estado)~., data = training,
   gamma =c(1:30)/150, cost=c(1:100)/5)
summary(obj3)
plot(obj3,main="Rendimiento del modelo de MSV", xlab="
   gamma", ylab="costo")
#Modelo de MSV
model3 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "radial",
   cost = 2.8, gamma=0.01333333)
summary(model3)
#Graficar el modelo de MSV
plot(model3, training, Jitter_rel~ Delta1)
#Clasificacion
clas3<-predict(model3,training)
table(clas3,training$Estado)
#Prediccion
pred3<-predict(model3,testing)
table(pred3,testing$Estado)

#n\'ucleo sigmoidal
#Funcion tune.svm para poder escojer los parametros
   optimos
obj4 <- tune.svm(factor(Estado)~., data = training,
   gamma =c(1:30)/150, cost=c(1:100)/5)
summary(obj4)
plot(obj4)
#Modelo de MSV
model4 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "sigmoid",
   cost = 2.8, gamma=0.01333333)
summary(model4)
#Graficar el modelo de MSV
plot(model4, training, Jitter_rel~ Delta1)
#Clasificacion
clas4<-predict(model4,training)
table(clas4,training$Estado)
#Prediccion
pred4<-predict(model4,testing)
table(pred4,testing$Estado)
```

```
#MODELO CON 11 VARIABLES
#Se cargan las librerias
library(lattice)
library(ggplot2)
library(caret)
library(e1071)
#Se cargan los datos
datos<-read.csv("C:/Users/ARTURO/Desktop/parkinson.csv",
    header= TRUE)
#Se dividen los datos aleatoriamente en datos de
   entrenamiento y de prueba -> 75%
#de los datos en datos de entrenamiento y 25% de los
   datos en datos de prueba
attach(datos)
names(datos)
set.seed(998)
inTraining <- createDataPartition(datos$Estado, p = .75,
    list = FALSE)
training <- datos[ inTraining, c
   (2,3,7,12,17,18,19,20,23,26,39,45)]
testing  <- datos[-inTraining, c
   (2,3,7,12,17,18,19,20,23,26,39,45)]
##################################################
#Hacemos el modelo de mÃ¡quinas de soporte vectorial
#Se prueban distintos n\'ucleo

#n\'ucleo lineal
#Funcion tune.svm para poder escojer los parametros
   optimos
obj2 <- tune.svm(factor(Estado)~., data = training,cost=
   c(1:200)/5)
summary(obj2)
plot(obj2,main="Rendimiento del modelo de MSV", xlab="
   costo", ylab="error")
#Modelo de MSV
model2 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "linear",
   cost = 0.6)
summary(model2)
#Graficar el modelo de MSV
plot(model2, training, Jitter_rel~ Delta11)
#Clasificacion
```

```
clas2<-predict(model2,training)
table(clas2,training$Estado)
#Prediccion
pred2<-predict(model2,testing)
table(pred2,testing$Estado)

#n\'ucleo polinomial
#Funcion tune.svm para poder escojer los parametros
   optimos
obj1 <- tune.svm(factor(Estado)~., data = training, cost
   =c(1:200)/5, degree = c(2:10))
summary(obj1)
plot(obj1, main="Rendimiento del modelo de MSV", xlab="
   grado", ylab="costo")
#Modelo de MSV
model1 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "polynomial",
    cost = 5.4 ,degree=2,coef.0=0)
summary(model1)
#Graficar el modelo de MSV
plot(model1, training, Jitter_rel~ Delta11)
#clasificacion
clas<-predict(model1,training)
table(clas,training$Estado)#clasifica mal 20
#Prediccion
pred<-predict(model1,testing)
table(pred,testing$Estado)

#n\'ucleo radial
#Funcion tune.svm para poder escojer los parametros
   optimos
obj3 <- tune.svm(factor(Estado)~., data = training,
   gamma =c(1:30)/150, cost=c(1:200)/5)
summary(obj3)
plot(obj3, main="Rendimiento del modelo de MSV", xlab="
   gamma", ylab="costo")
#Modelo de MSV
model3 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "radial",
   cost = 0.8, gamma=0.08)#0.8,0.08
summary(model3)
#Graficar el modelo de MSV
plot(model3, training, Jitter_rel~ Delta11)
```

```
#Clasificacion
clas3<-predict(model3,training)
table(clas3,training$Estado)
#Prediccion
pred3<-predict(model3,testing)
table(pred3,testing$Estado)

#n\'ucleo sigmoidal
#Funcion tune.svm para poder escojer los parametros
   optimos
obj4 <- tune.svm(factor(Estado)~., data = training,
   gamma =c(1:30)/50, cost=c(1:100)/5)
summary(obj4)
plot(obj4)
#Modelo de MSV
model4 <- svm(factor(Estado) ~ ., data = training,
   method = "C-classification", n\'ucleo = "sigmoid",
   cost = 0.8, gamma=0.08)
summary(model4)
#Graficar el modelo de MSV
plot(model4, training, Jitter_rel~ Delta11)
#Clasificacion
clas4<-predict(model4,training)
table(clas4,training$Estado)
#Prediccion
pred4<-predict(model4,testing)
table(pred4,testing$Estado)
```

Bibliografía

Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge.

Gutiérrez, J. (2017). Aplicación del modelo de regresión logística en mediciones de registros vocales para el diagnóstico de la enfermedad de parkinson. Master's thesis, UNAM, Facultad de Ciencias.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning with applications in R*. Springer.

Kecman, V. (2005). Support vector machines - an introduction. In Wang, L., editor, *Suport Vector Machines: Theory and Applications*, Studies in Fuzzines and Soft Computing, pages 1–47. Springer.

Naranjo, L., Pérez, C., Campos-Roca, Y., and Martín, J. (2016). Addresing voice recording replications for Parkinson's disease detection. *Expert Systems with Applications*, 46:286–292.

Osuna, E., Freund, R., and Girosi, F. (1997). Support vector machines: Trainig and applications. Technical Report 144, Massachusetts Institute of Technology Artificial Intelligence Laboratory and Center for Biological and Computation Learning Department of Brain and Cognitive Sciences.

Pérez, C., Campos-Roca, Y., Naranjo, L., and Martín, J. (2016). Diagnosis and tracking of Parkinson's disease by using automatically extracted acoustic features. *Journal of Alzheimer's Disease & Parkinsonism*, 6(5):260.

Shigeo, A. (2010). *Support Vector Machines for Pattern Classification*. Springer, 2nd edition.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley.