Centro de Investigación en Matemáticas (CIMAT). Unidad Monterrey

Ciencia de datos 3

Jorge Luis Ramos Zavaleta

3 de Mayo de 2018

1. Ejercicio

Este ejercicio es sobre la evaluación de diferentes algoritmos de clasificación binaria.

Tenemos un conjunto de prueba y un conjunto de entrenamiento y dos algoritmos de clasificación C_1 y C_2 obtenidos con el mismo conjunto de entrenamiento. Define X_i como la variable que indica si el algoritmo C_i clasifica un dato del conjunto de prueba como 0 o 1 y define Y_i como la variable que indica si el algoritmo C_i lo clasifica correctamente o no. Lo anterior nos da dos tablas de contingencia (una basada en (X_1, X_2) y otra en (Y_1, Y_2)).

1. Una manera para cuantificar si los clasificadores se comportan de manera similar es verificar si las distribuciones marginales de Y_i son iguales. Muestra que lo anterior es equivalente a verificar si $p_{0,1} = p_{1,0}$ con p las probabilidades subyacentes a la tabla de contingencia de (Y_1, Y_2). Deriva el estimador de Máxima Verosimilitud para esta hipótesis. Usa un estadístico de prueba basado en razones de verosimilitud para calcular $p-$valor bajo esta hipótesis para los siguientes datos:

	$Y_2 = 0$	$Y_2 = 1$
$Y_1 = 0$	8	7
$Y_1 = 1$	11	21

2. ¿Qué complicaciones habría al utilizar las X's en lugar de las Y's?

1.1. Solución

1. Suponiendo que los resultados obtenidos por los clasificadores son independientes entonces

$$p_{01} = P(Y_1 = 0, Y_2 = 1) = P(Y_1 = 0)P(Y_2 = 1) =$$
$$= (1 - P(Y_1 = 1))(1 - P(Y_2 = 0)) = 1 - P(Y_2 = 0) - P(Y_1 = 1) + P(Y_1 = 1)P(Y_2 = 0) =$$
$$= 1 - P(Y_2 = 0) - P(Y_1 = 1) + p_{10}$$

Luego $P_{01} = p_{10}$ si y solo si $1 - P(Y_2 = 0) - P(Y_1 = 1) = 0$. Ahora

$$\begin{aligned} 1 - P(Y_2 = 0) - P(Y_1 = 1) &= (1 - P(Y_2 = 0)) - P(Y_1 = 1) \\ &= P(Y_2 = 1) - P(Y_1 = 1) \end{aligned}$$

o bien haciendo las manipulaciones algebraicas al revés $P(Y_2 = 0) - P(Y_1 = 0)$, por lo que para que estas expresiones sean cero se requiere que las distribuciones marginales de Y_1 y Y_2 sean iguales. Así $p_{01} = p_{10}$ si y solo si las distribuciones marginales de Y_1 y Y_2 coinciden.

Ahora para esta hipótesis $p = p_{01} - p_{10}$ es el estimador que queremos obtener, siendo que C_1 y C_2 son clasificadores binarios entonces p puede considerarse una variable binomial, y su estimador de maxima verosimilitud esta dado por la razón entre el número de casos favorables y el número de veces que se repitió el evento. Por otro se utilizo un estadístico de prueba basado en razones de verosimilitud conocido como G-test que esta dado por

$$G = 2ln\frac{P(X = k|H_0)}{P(X = k|H_1)}$$

el cual se distribuye como una chi cuadrada con (numero de filas-1)(numero de columnas-1) grados de libertad.

El estadístico también se expresa de la siguiente manera

$$G = 2\left(\sum_{i=1}^{r}\sum_{j=1}^{c} O_{ij} ln\left(\frac{O_{ij}}{E_{ij}}\right)\right)$$

donde r y c son los tamaños de las filas y las columnas, O_{ij} es el conteo observado en la fila i columna j y E_{ij} esta dado de la siguiente manera

$$E_{ij} = \frac{(suma\ fila\ i)(suma\ columna\ j)}{suma\ de\ la\ tabla}$$

Para hacer más fácil el cálculo se genero la tabla a usar por los E_{ij} y quedo de la siguiente manera

	$Y_2 = 0$	$Y_2 = 1$	
$Y_1 = 0$	6	9	15
$Y_1 = 1$	13	19	32
	19	28	47

por lo que nuestro estadístico queda de la siguiente manera

$$G = 1.5106$$

y el p-valor es

$$p - valor = 1 - \chi^2_1(1.5106) = 0.2190$$

2. Las variables X's pueden no ser independientes ya que un clasificador puede estar basado en el otro por lo que el trabajar con la distribución conjunta puede ser mucho peor dado que la distribución conjunta no puede verse simplemente como la multiplicación de las marginales y su calculo puede no ser algo simple.

2. Ejercicio

Para datos de clasificación binaria $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, considera la siguiente función de costo:

$$\mathcal{L} = \sum_i (\theta(y_i) - \beta' \mathbf{x}_i - \beta_0)^2 \tag{2.1}$$

Definimos n_+, n_- el número de observaciones con $y_i = 1$ y $y_i = -1$, respectívamente $\mathbf{c}_+, \mathbf{c}_-$ el centroide de las observaciones con $y_i = 1$, y $y_i = -1$ y $\mathbf{c}$ el centroide de todos los datos. Como en clase, construimos las matrices:

$$\begin{aligned} \mathbf{S}_B &= (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)' \\ \mathbf{S}_W &= \textstyle\sum_{i:y_i=1}(\mathbf{x}_i - \mathbf{c}_+)(\mathbf{x}_i - \mathbf{c}_+)' + \sum_{i:y_i=-1}(\mathbf{x}_i - \mathbf{c}_-)(\mathbf{x}_i - \mathbf{c}_-)' \end{aligned}$$

1. Verifica que

$$\mathbf{S}_W = \sum_{i:y_i=1} \mathbf{x}_i \mathbf{x}_i' + \sum_{i:y_i=-1} \mathbf{x}_i \mathbf{x}_i' - n_+ \mathbf{c}_+ \mathbf{c}_+' - n_- \mathbf{c}_- \mathbf{c}_-'$$

2. Verifica que el vector $\mathbf{S}_B \boldsymbol{\beta}$, es un múltiplo del vector $(\mathbf{c}_+ - \mathbf{c}_-)$.

3. Si definimos $\theta(1) = n/n_+$ y $\theta(-1) = -n/n_-$, verifica que en el mínimo de (2.1):

$$\beta_0 = -\boldsymbol{\beta}' \mathbf{c},$$

$$(\mathbf{S}_W + \frac{n_+ n_-}{n} \mathbf{S}_B)\boldsymbol{\beta} = n(\mathbf{c}_+ - \mathbf{c}_-) \tag{2.2}$$

4. Usando el resultado de inciso b, argumenta que (2.2) implica que en el mínimo:

$$\boldsymbol{\beta} \sim S_W^{-1}(\mathbf{c}_+ - \mathbf{c}_-),$$

es decir la solución coincide con la del Fisher Discriminant Analysis (FDA).

5. Lo anterior permite implementar FDA usando algún algoritmo de mínimos cuadrados. En `R` lo haremos con la función `lm()`. Ilustra cómo funciona el método con algunos conjuntos de datos en 2D bien elegidos.

6. Observamos que (2.1) muestra que FDA no es muy robusto a datos atípicos.

   Una posibilidad para hacerlo más robusto es usar mínimos cuadrados ponderados. Por ejemplo `lm()` tiene un argumento opcional `weights` donde se pueden proporcionar pesos $w_i, i = 1, \ldots, n$ para minimizar:

$$\sum_i w_i(\theta(y_i) - \boldsymbol{\beta}^t \mathbf{x}_i - \beta_0)^2.$$

   ¿Cóomo elegirías estos pesos? Verifica tu propuesta con algunos ejemplos en 2D.

2.1. Solución

1. Primero observemos que

$$\sum_{i:y_i=1} x_i = n_+ c_+ \quad y \quad \sum_{i:y_i=-1} x_i = n_- c_-$$

   Ahora

$$\begin{aligned}
S_W &= \sum_{i:y_i=1} (x_i - c_+)(x_i - c_+)^t + \sum_{i:y_i=-1} (x_i - c_-)(x_i - c_-)^t \\
&= \sum_{i:y_i=1} x_i x_i^t - \sum_{i:y_i=1} (x_i c_+^t + c_+ x_i^t) + \sum_{i:y_i=-1} c_+ c_+^t + \sum_{i:y_i=-1} x_i x_i^t - \sum_{i:y_i=-1} (x_i c_-^t + c_- x_i^t) + \sum_{i:y_i=-1} c_- c_-^t \\
&= \sum_{i:y_i=1} x_i x_i^t - \sum_{i:y_i=1} (x_i c_+^t + (x_i c_+^t)^t) + n_+ c_+ c_+^t + \sum_{i:y_i=-1} x_i x_i^t - \sum_{i:y_i=-1} (x_i c_-^t + (x_i c_-^t)^t) + n_- c_- c_-^t \\
&= \sum_{i:y_i=1} x_i x_i^t - 2n_+ c_+ c_+^t + n_+ c_+ c_+^t + \sum_{i:y_i=-1} x_i x_i^t - 2n_- c_- c_-^t + n_- c_- c_-^t \\
&= \sum_{i:y_i=1} x_i x_i^t - n_+ c_+ c_+^t + \sum_{i:y_i=-1} x_i x_i^t - n_- c_- c_-^t
\end{aligned}$$

2. Tenemos que $S_B = (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)^t$ por lo que $S_B\beta = (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)^t\beta = (\mathbf{c}_+ - \mathbf{c}_-)k$ con k un escalar. Luego $S_B\beta$ es un múltiplo de $(\mathbf{c}_+ - \mathbf{c}_-)$.

3. Derivando la función de costos con respecto de β_0 y β e igualando a cero tenemos

$$-2\sum_i(\theta(y_i)-\beta^t x_i-\beta_0)=0$$

y

$$-2\sum_i(\theta(y_i)-\beta^t x_i-\beta_0)xi^t=0$$

respectivamente. Ahora notese que

$$\sum_i\theta(y_i)=\sum_{i:y_i=1}\theta(y_i)+\sum_{i:y_i=-1}\theta(y_i)=n_+\frac{n}{n_+}+n_-\frac{-n}{n_-}=0$$

por lo que de la primera ecuación tenemos que

$$\begin{aligned}\sum_i-\beta^t xi&=\sum_i\beta_0\\ -\beta^t nc&=n\beta_0\\ \beta 0&=-\beta^t c\end{aligned}$$

Usando este resultado en la segunda ecuación tenemos

$$\begin{aligned}\sum_i\theta(y_i)xi^t-\sum_i\beta^t x_i xi^t-\sum_i\beta_0 xi^t&=0\\ \sum_{i:y_i=1}\frac{n}{n_+}xi^t-\sum_{i:y_i=1}\frac{n}{n_-}xi^t-\sum_i\beta^t x_i xi^t-\sum_i\beta_0 xi^t&=0\\ n(\frac{1}{n_+}\sum_{i:y_i=1}xi^t-\frac{1}{n_-}\sum_{i:y_i=1}xi^t)-\beta^t\sum_i x_i xi^t-\beta_0\sum_i xi^t&=0\\ n(c_+-c_-)^t&=\beta^t\sum_i x_i xi^t+\beta_0\sum_i xi^t\\ n(c_+-c_-)^t&=\beta^t\sum_i x_i xi^t-\beta^t c\sum_i x_i^t\\ n(c_+-c_-)^t&=\beta^t\left(\sum_i x_i xi^t-c\sum_i xi^t\right)\\ n(c_+-c_-)^t&=\beta^t\left(\sum_i x_i^t xi-c^t\sum_i xi\right)^t\\ n(c_+-c_-)&=(S_W+n_+c_+^t c_+ +n_-c_-^t c_- -ncc^t)\beta\\ n(c_+-c_-)&=\left(S_W+n_+c_+^t c_+ +n_-c_-^t c_- -(n_+c_+ +n_-c_-)c^t\right)\beta\\ n(c_+-c_-)&=\left(S_W+n_+c_+c_+^t+n_-c_-c_-^t-(n_+c_+ +n_-c_-)\frac{(n_+c_+ +n_-c_-)^t}{n}\right)\beta\\ n(c_+-c_-)&=\left(S_W+\frac{n_+n_-(c_+c_+^t+c_-c_-^t+c_+c_-^t+c_-c_+^t)}{n}\right)\beta\end{aligned}$$

Luego

$$n(c_+ - c_-) = \left(S_W + \frac{n_+ n_- S_B}{n} \right) \beta$$

4. Dado que en el ejercicio anterior obtuvimos que

$$n(c_+ - c_-) = \left(S_W + \frac{n_+ n_- S_B}{n} \right) \beta$$

y del ejercicio 2 sabemos que $S_B\beta$ es un múltiplo de $(c_+ - c_-)$. Entonces

$$\begin{aligned} n(c_+ - c_-) &= \left(S_W + \frac{n_+ n_- S_B}{n} \right) \beta \\ n(c_+ - c_-) &= S_W\beta + \frac{n_+ n_-}{n} S_B\beta \\ n(c_+ - c_-) &= S_W\beta + \frac{n_+ n_-}{n} k(c_+ - c_-) \end{aligned}$$

Luego

$$S_w\beta = \frac{n_+ n_- k + n^2}{n}(c_+ - c_-)$$

Por lo que

$$\beta = S_w^{-1} \frac{n_+ n_- k + n^2}{n}(c_+ - c_-) \sim S_w^{-1}(c_+ - c_-)$$

5. Para el caso de mínimos cuadrados si se escalan los datos tedríamos que resolver el problema $Ax = b$ con

$$b = \begin{pmatrix} 1/n_+ \\ 1/n_+ \\ \vdots \\ -1/n_- \\ -1/n_- \\ \vdots \\ -1/n_- \end{pmatrix}, \quad A = \begin{pmatrix} x_1^t \\ x_2^t \\ \vdots x_{n_+}^t \\ x_{n_+ + 1}^t \\ \vdots \\ x_{n_+ + n_-}^t \end{pmatrix}$$

Para el primer ejemplo se consideraron dos clases diferidas en 5 y 6 puntos en $\mathbb{R}^2$ tal y como se muestran en las figuras 2.1 y 2.2. Se puede observar que la recta no genera una buena separación, lo cual puede deberse a la constante que se pierde al final del inciso anterior para ajustar la forma a una regresión lineal. Para el segundo caso se considera conjuntos de datos generados a partir de distribuciones normales de manera aleatoria, lo cual parece ayudar mucho más a generar una buena recta de separación como puede observarse en las figuras 2.3 y 2.4.

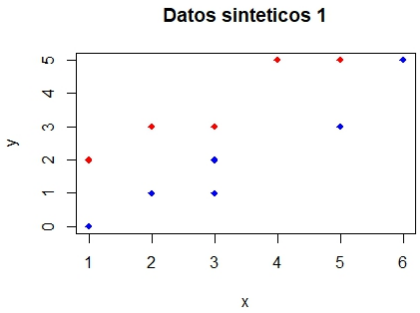

Figura 2.1: Datos sin escalar

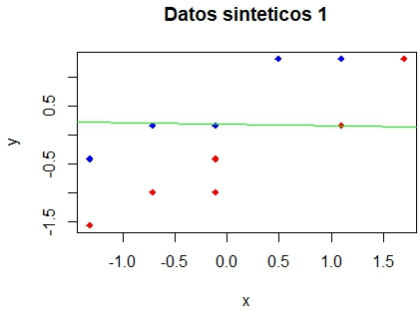

Figura 2.2: Datos escalados con la recta de separación aplicando el método de Fisher para clasificar

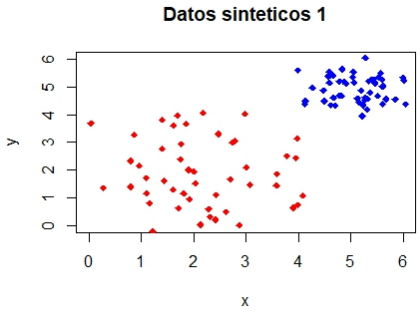

Figura 2.3: Datos sin escalar

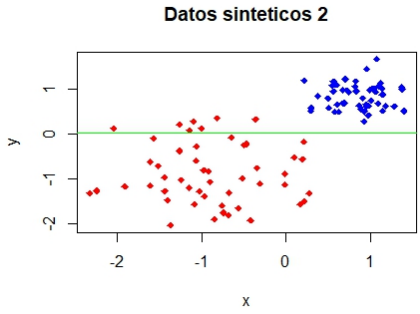

Figura 2.4: Datos escalados con la recta de separación aplicando el método de Fisher para clasificar

6. Para el siguiente ejemplo se agrego un punto extra atípico como se muestra en la figura 2.5 y se considero usar mínimos cuadrados pesados para disminuír la influencia que tiene el punto sobre el circuncentro de los datos en su clase. Para ello los demás pesos se pusieron como 1, mientras que el peso respectivo para el dato atípico se obtuvo obteniendo la razón entre la distancia del punto atípico al circuncentro de los datos de su clase entre la distancia de dicho punto al al circuncentro de los datos de su clase sin considerar el dato atípico y se obtuvo un resultado similar al anterior como se puede observar en la figura 2.6.

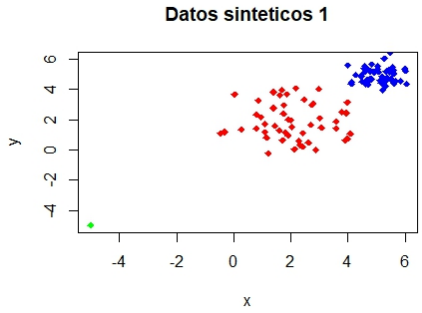

Figura 2.5: Datos sin escalar con el dato atípico de la clase roja coloreado en verde para

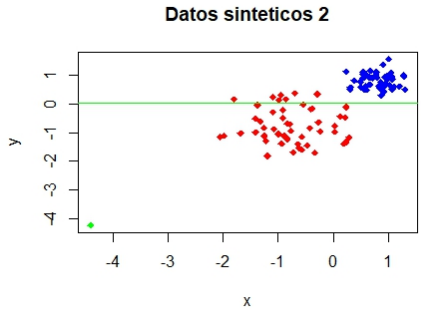

Figura 2.6: Datos escalados con la recta de separación aplicando el método de Fisher para clasificar

3. Ejercicio

Este ejercicio es sobre el método de clasificación binaria perceptron.

1. Implementa el modelo clásico de perceptrón (versión que trabaja en línea). Aplícalo primero a un conjunto de datos artificiales en dos dimensiones y con dos categorias. Discute tus resultados. Incluye gráficas del ajuste.

2. Aplica el clasificador al conjunto de datos `pima` que vimos en clase. Usa `pima.tr` para ajustar el modelo y `pima.te` para verificar su calidad predictiva. ¿Qué puedes decir sobre su desempeño? Comenta tus hallazgos.

3.1. Solución

La implementación del algoritmo se detalla en los archivos adjuntos a este documento. Para probar el algoritmo perceptron se generaron 1 conjuntos de datos divido en 2 clases. La primera clase consta de 100 puntos en $\mathbb{R}^2$ 50 generados aleatoriamente por una distribución normal univariada con media 1 y desviación estándar 1 y de 50 puntos de una distribución normal univariada con media 3 y desviación estándar 0.5, la clase 2 esta compuesta solo por 50 puntos generados con una normal univariada con media 6 y desviación estándar 1. La distribución original en clases se puede ver en la figura 3.1 y la clasificación alcanzada con el algoritmo perceptron en linea se puede observar en la figura 3.2.

En este sentido puede observarse de ambas figuras que la clasificación se logra bastante bien, aunque los datos son linealmente separables y por tanto no es algo sorprendente que el algoritmo logre una buena clasificación. Sin embargo el nivel de clasificación fue del 100 % lo cual es bastante admirable para ser un algoritmo tan simple. Este último hecho puede constatarse en la siguiente tabla

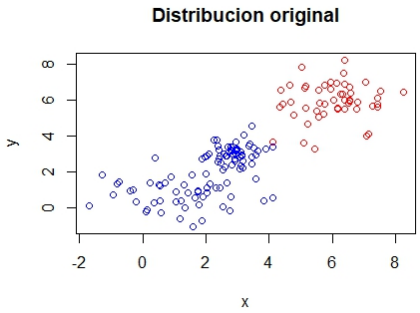

Figura 3.1: Distribución original de las clases

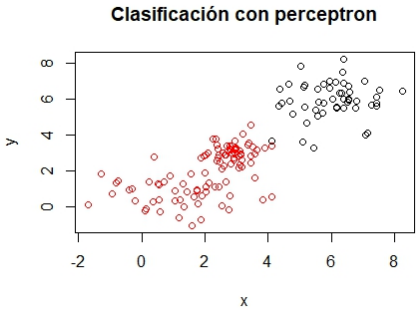

Figura 3.2: Distribución lograda al clasificar usando el algoritmo perceptron en linea

	Etiquetas	
Clasificación	-1	1
-1	100	0
1	0	50

Para la siguiente parte se utilizaron los datos pima correspondientes a un diagnostico de diabetes en mujeres de la tribu pima mayores de 21 años bajo diferentes variables como nivel de insulina y edad. Se tuvo que hacer un pequeño preprocesado ya que la columna de etiquetas en los datos pima de entrenamiento eran númericos y en los datos de prueba eran de tipo caracter, aparte de que el encoding número era de 0 y 1 inicialmente y se cambio a 1 y -1 para que funcionara con la clasificación del perceptron.

Se observo que con 220 iteraciones utilizadas el algoritmo perceptron no converge para este conjunto de datos, y el aumentar el número de iteraciones tiende a empeorar la clasificación. La clasificación obtenida para los datos de entrenamiento y prueba se puede observar en las siguientes tablas y de donde se puede deducir que se obtuvo una precisión en el entrenamiento de 71.3 % y de 71 % en los datos de prueba.

Entrenamiento		
	-1	1
-1	207	79
1	16	30

Prueba		
	-1	1
-1	124	50
1	8	18

4. Ejercicio

Implementa Kernel FDA. Puedes basarte en el artículo de Mika et al: *Fisher Discriminant Analysis with kernels*. Verifica su desempeño en un conjunto de datos artificiales *apropiados* en dos dimensiones y dos clases. Compara su desempeño con FDA estándar en el conjunto de datos `pima` del ejercicio 2. Comenta tus hallazgos.

Opcional (puntos extra): realiza la implementación como una librería en `C` o `C++` para `R`. Incluye la interfaz (o wrapper) adecuado en `R` para usarse.

4.1. Solución

Para la implementación del algoritmo de kernel FDA se requería generar una matriz de Gram muy grande por lo que se implemento una versión haciendo uso de *Rcpp* con lo que el calculo de la matriz de Gram es cerca de unas 7 veces más rápido que hacerlo directamente con R, la otra parte del codigo se realizó en *R* por simplicidad el codigo se adjunta a este documento en un archivo para permitir su reproducibilidad.

Para este ejercicio se debia probar el algoritmo kernel FDA contra su versión lineal sobre el conjuntos de datos *pima*. Pero para asegurar que su funcionamiento era el correcto se utilizo un conjunto de datos correspondiente a una sigmoíde con el fin de ver si el algoritmo podía clasificar suficientemente bien sus puntos. En la figura se puede observar el conjunto con su clasificación original y en la figura se tiene la figura obtenida usando KFDA con kernel gaussiano con un parámetro de 0.074 y un shift de desplazamiento para permitir que N sea invertible de 0.01 con lo que se tuve un 93.25 % de eficiencia en la clasificación.

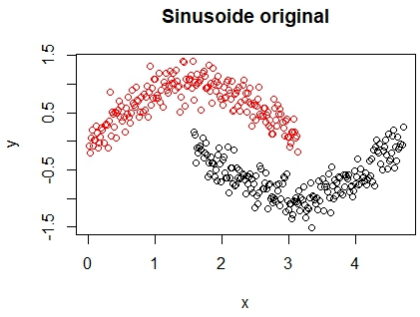

Figura 4.1: Sigmoide con sus cluster originales

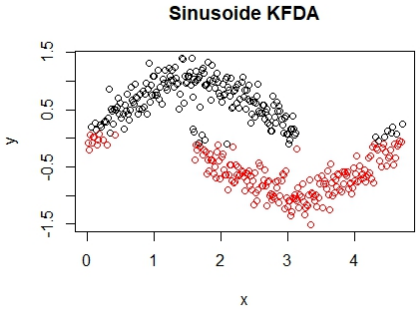

Figura 4.2: Clasificación del sigmoide usando un kernel gaussiano con parámetro de 0.074

Cabe destacar que el uso del shift también puede generar variación en la clasificación por lo que se eligió el más pequeño posible para que no se alteren los resultados de la matriz N

original. Para la segunda parte se hizo uso del conjunto pima que tiene algunas variables como nivel insulina, edad y BMI para clasificar pacientes con diabetes el estudio se realizo con datos de mujeres mayores de 21 años pertenecientes a la tribu pima.

Se aplico el algoritmo KFDA con kernel gaussiano con un parámetro de 0.002 y un shift de 0.01 a los datos pima y se obtuvo una precisión de 71.5 %, pero haciendo uso de LDA con una distribución prior uniforme se obtuvo una eficiencia mayor en este caso de 76.5 %, aunque la búsqueda del parámetro de ajuste óptimo podría tal vez aumentar un poco la precisión del algoritmo KFDA. La implementación se detalla en el anexo.

5. Ejercicio

Los archivos contenidos en las carpetas `email_train` e `email_test` corresponden a correos electrónicos en inglés clasificados como Spam y No-Spam.

1. Implementa clasificadores de Spam usando regresión logística, LDA, QDA, FDA, Kernel FDA y redes neuronales. Usa los datos `email_train` e `email_test` para ajustar y probar los métodos, respectívamente. Compara su desempeño.

2. Las curvas ROC (Receiver Operating Characteristics) es un método muy común para comparar algoritmos de clasificación binarios basado en la tabla de errores (falsos positivos y falsos negativos) que se cometen. Usa los resultados del inciso anterior para comparar los clasificadores usando este criterio. ¿Cuál método elegirias? Usa el criterio del área bajo la curva (AUC).

Hay bastante literatura sobre ROC, como referencia, puedes consultar el paper de T. Fawcett, An Introduction to ROC Analysis. En `R`, la libreria `pROC` puede ser útil.

5.1. Solución

Para este ejercicio se utilizaron los conjuntos de datos *emailtrain* e *emailtest* que contienen correos electrónicos en inglés etiquetados como Spam y no-Spam, en el primer caso se cuenta con 2200 correos de los cuales 1400 son etiquetados como Spam y 800 como no-Spam. Se entrenaron varios algoritmos con los datos de entrenamiento para comprobar su eficiencia al clasificar un conjunto de prueba que contiene 500 correos etiquetados como Spam y otros 500 etiquetados como no-Spam.

Para trabajar con el conjunto de datos se consideraron las 40 palabras más frecuentes y se tuvo que hacer un pequeño procesamiento extra al recomendado para deshacernos de caracteres raros que no pertenecen al idioma inglés y que restringían la creación la matriz de documentos.

La presición se tomó conconsiderando el ajuste que se logra con las etiquetas de Spam y no-Spam, en este sentido cabe destacar que los algoritmos de redes neuronales tienen un factor probabbilístico por lo que se fijo una semilla para mantener su reproducibilidad, así como también para el kernel FDA se considero un kernel gaussiano con parámetro 0.05 y shift de 0.01, también se tuvo que agregar un poco de ruído al conjunto de datos para permitir que se cumplieran las condiciones de funcionamiento del algoritmo QDA. Una última cuestion es que se genero un sobreaajuste con la regresión logística debido a que algunas de las columnas solo tenían entradas binarias como la columna de etiquetas.

Los resultados de precisión para los datos de prueba se resumen en la siguiente tabla

Método	Precisión
Regresión logística	83.5 %
LDA	76.5 %
QDA	73.9 %
FDA	77.1 %
Kernel FDA	82.9 %
Red neuronal con 3 nodos y 1 capa escondida	83.9 %
Red neuronal con 5 nodos y 1 capa escondida	86.5 %

En este sentido cualquiera elegiría el método que mayor eficiencia tiene sin embargo se pueden considerar otros enfoques como las curvas ROC que nos dan una relación entre los falsos positivos y los verdaderos positivos, en este sentido si definimos el treshold entre 0 y 1, nos interesa en términos del área bajo la curva (AUC) que sea lo más cercana posible a 1. En nuestro caso se obtuvieron los siguientes valores AUC para cada clasificador

Clasificador	AUC
Regresión logística	0.8551
LDA	0.8323
QDA	0.8067
FDA	0.7776
Kernel FDA	0.7592
Red neuronal con 3 nodos y 1 capa escondida	0.8402
Red neuronal con 5 nodos y 1 capa escondida	0.836

Mientras que en términos de precisión uno elegiría una red neuronal con 5 nodos y 1 capa escondida, en términos del área bajo la curva ROC se elegiría la regresión logística o la red neuronal con 3 nodos y una capa escondida, que sería preferible usar ésta última debido al sobreajuste que presento la regresión logística.

Se intentaron buscar otras variables para generar la clasificación como el número de carácteres en los mails con Spam y sin Spam, y se buscaron las palabras *www* y *prize* entre los

correos para colocarlos como spam pero tanto en los correos de Spam y no-Spam hay mucha variabilidad en su aparación por lo que ninguno de estos críterios nos generó variables para lograr una buena clasificación de los correos.