Centro de Investigación en Matemáticas (CIMAT). Unidad Monterrey

Tarea 5

Enrique Santibáñez Cortés

11 de mayo de 2021

1. Clasificación binaria

1.1. Introducción

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

$$L = \sum_{i=1}^{n} \big(\theta(y_i) - \beta'\mathbf{x}_i - \beta_0\big)^2 \tag{1.1}$$

Definamos 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 $y_1 = -1$ y $\mathbf{c}$ el centroide de todos los datos. Y definamos las siguientes matrices:

$$\mathbf{S}_B = (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)' \tag{1.2}$$

$$\mathbf{S}_W = \sum_{i:y_i=1} (\mathbf{x}_i - \mathbf{c}_+)(\mathbf{x}_i - \mathbf{c}_+)' + \sum_{i:y_i=-11} (\mathbf{x}_i - \mathbf{c}_-)(\mathbf{x}_i - \mathbf{c}_-)' \tag{1.3}$$

1.2. Propiedades de las matrices $\mathbf{S}_B$ y $\mathbf{S}_W$

Propiedad: 1.

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

Demostración de la Propiedad: 1. Recordemos que $\mathbf{c}_+ = \frac{1}{n_+}\sum_{i:y_i=1} x_i$ y $\mathbf{c}_- = \frac{1}{n_-}\sum_{i:y_i=-1} x_i$. Ahora simplemente desarrollamos los productos de la definición de $\mathbf{S}_W$ (1.2),

$$\mathbf{S}_W = \sum_{i:y_i=1} (\mathbf{x}_i - \mathbf{c}_+)(\mathbf{x}_i - \mathbf{c}_+)' + \sum_{i:y_i=-11} (\mathbf{x}_i - \mathbf{c}_-)(\mathbf{x}_i - \mathbf{c}_-)'$$

$$
\begin{aligned}
&= \sum_{i:y_i=1} \left(\mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_+\mathbf{x}_i' + \mathbf{c}_+\mathbf{c}_+'\right) + \sum_{i:y_i=-1} \left(\mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_-\mathbf{x}_i' - \mathbf{c}_-\mathbf{c}_-'\right) \\
&= \sum_{i:y_i=1} \mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_+ \sum_{i:y_i=1} \mathbf{x}_i' + n_+\mathbf{c}_+\mathbf{c}_+' + \sum_{i:y_i=-1} \mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_- \sum_{i:y_i=-1} x_i' - n_-\mathbf{c}_-\mathbf{c}_-' \\
&= \sum_{i:y_i=1} \mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_+(n_+\mathbf{c}_+') + n_+\mathbf{c}_+\mathbf{c}_+' + \sum_{i:y_i=-1} \mathbf{x}_i\mathbf{x}_i' - 2\mathbf{c}_-(n_-\mathbf{c}_-) - n_-\mathbf{c}_-\mathbf{c}_-' \\
&= \sum_{i:y_i=1} \mathbf{x}_i\mathbf{x}_i' + \sum_{i:y_i=-1} \mathbf{x}_i\mathbf{x}_i' - n_+\mathbf{c}_+\mathbf{c}_+' - n_-\mathbf{c}_-\mathbf{c}_-' \ \blacksquare.
\end{aligned}
$$

Propiedad: 2. *El vector $\mathbf{S}_B\beta$ es un múltiplo del vector $(\mathbf{c}_+ - \mathbf{c}_-)$.*

Demostración de la Propiedad: 2. Podemos definir a $\mathbf{m}_+ = \frac{1}{n_+}\sum_{i:y_i=1} \beta' x_i$ y $\mathbf{m}_- = \frac{1}{n_-}\sum_{i:y_i=-1} \beta' x_i$, ya que son productos puntos de dos vectores podemos decir que $\mathbf{m}_+$ y $\mathbf{m}_-$ son escalares. Entonces ocupando la definición de $\mathbf{S}$, tenemos que

$$
\begin{aligned}
\mathbf{S}_B\beta &= (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)'\beta = (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+\beta' - \mathbf{c}_-\beta')' \\
&= (\mathbf{c}_+ - \mathbf{c}_-)\left(\frac{1}{n_+}\sum_{i:y_i=1} \beta' x_i - \frac{1}{n_-}\sum_{i:y_i=-1} \beta' x_i\right)' \\
&= (\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{m}_+ - \mathbf{m}_-)' = (\mathbf{c}_+ - \mathbf{c}_-)\gamma,
\end{aligned}
$$

donde $\gamma = \left(\frac{1}{n_+}\sum_{i:y_i=1} \beta' x_i - \frac{1}{n_-}\sum_{i:y_i=-1} \beta' x_i\right)'$. Por lo tanto, podemos decir que $\mathbf{S}_B\beta$ es un múltiplo del vector $(\mathbf{c}_+ - \mathbf{c}_-)$ $\blacksquare$.

Propiedad: 3. *Sea $\theta(1) = \frac{n}{n_+}$ y $\theta(-1) = -\frac{n}{n_-}$, tenemos que el mínimo de (1.1) es*

$$
\begin{aligned}
\beta_0 &= -\beta'\mathbf{c}. \\
\left(\mathbf{S}_W + \frac{n_+n_-}{n}\mathbf{S}_B\right)\beta &= n(\mathbf{c}_+ - \mathbf{c}_-).
\end{aligned}
$$

Demostración de la Propiedad: 3. La función de costo se puede reescribir como

$$
\begin{aligned}
\sum_{i=1}^{n} \left(\theta(y_i) - \beta'\mathbf{x}_i - \beta_0\right)^2 &= \sum_{i:y_i=1} \left(\theta(y_i) - \beta'\mathbf{x}_i - \beta_0\right)^2 + \sum_{i:y_i=-1} \left(\theta(y_i) - \beta'\mathbf{x}_i - \beta_0\right)^2 \\
&= \sum_{i:y_i=1} \left(\frac{n}{n_+} - \beta'\mathbf{x}_i - \beta_0\right)^2 + \sum_{i:y_i=-1} \left(-\frac{n}{n_-} - \beta'\mathbf{x}_i - \beta_0\right)^2.
\end{aligned}
$$

Ahora encontramos el mínimo de la expresión anterior para β_0, para ello primero derivamos respecto a β_0:

$$
\frac{\partial L}{\partial \beta_0} = -2\sum_{i:y_i=1} \left(\frac{n}{n_+} - \beta'\mathbf{x}_i - \beta_0\right) - 2\sum_{i:y_i=-1} \left(-\frac{n}{n_-} - \beta'\mathbf{x}_i - \beta_0\right).
$$

Igualamos a cero la expresión anterior y despejamos a β_0,

$$\begin{aligned}
-2\sum_{i:y_i=1}\left(\frac{n}{n_+}-\beta'\mathbf{x}_i-\beta_0\right)-2\sum_{i:y_i=-1}\left(-\frac{n}{n_-}-\beta'\mathbf{x}_i-\beta_0\right) &= 0\\
\frac{nn_+}{n_+}-\beta'\sum_{i:y_i=1}\mathbf{x}_i-n_+\beta_0-\frac{nn_-}{n_-}-\beta'\sum_{i:y_i=-1}\mathbf{x}_i-n_-\beta_0 &= 0\\
-\beta_0(n_++n_-) &= \beta'\left(\sum_{i:y_i=1}\mathbf{x}_i+\sum_{i:y_i=-1}\mathbf{x}_i\right)\\
\beta_0 &= -\beta'\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\\
\beta_0 &= -\beta'\mathbf{c}\ \blacksquare.
\end{aligned}$$

Realizando un procedimiento similar al anterior, derivamos la función de costo respecto a β

$$\frac{\partial L}{\partial \beta}=-2\sum_{i:y_i=-1}\left(\frac{-n}{n_-}-\beta'\mathbf{x}_i-\beta_0\right)\mathbf{x}_i'-2\sum_{i:y_i=1}\left(\frac{n}{n_+}-\beta'\mathbf{x}_i-\beta_0\right)\mathbf{x}_i'.$$

Ahora igualamos a cero e intentamos β',

$$\begin{aligned}
-2\sum_{i:y_i=1}\left(\frac{-n}{n_-}-\beta'\mathbf{x}_i-\beta_0\right)\mathbf{x}_i'-2\sum_{i:y_i=-1}\left(\frac{n}{n_+}-\beta'\mathbf{x}_i-\beta_0\right)\mathbf{x}_i' &= 0\\
\frac{-n\sum_{i:y_i=-1}\mathbf{x}_i}{n_-}-\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'-\beta_0\sum_{i:y_i=-1}\mathbf{x}_i'+\frac{n\sum_{i:y_i=1}\mathbf{x}_i}{n_+}-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'-\beta_0\sum_{i:y_i=1}\mathbf{x}_i' &= 0\\
-n\mathbf{c}_--\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'-\beta_0\sum_{i:y_i=-1}\mathbf{x}_i'+n\mathbf{c}_+-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'-\beta_0\sum_{i:y_i=1}\mathbf{x}_i' &= 0\\
-\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'-\beta_0 n_-\mathbf{c}_-'+-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'-\beta_0 n_+\mathbf{c}_+ &= n\mathbf{c}_--n\mathbf{c}_+\\
-\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'+n_-\beta'\mathbf{c}\mathbf{c}_-'-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'+n_+\beta'\mathbf{c}\mathbf{c}_+' &= n(\mathbf{c}_--\mathbf{c}_+).
\end{aligned}$$

Ocupando que $\mathbf{c}=\frac{n_+\mathbf{c}_++n_-\mathbf{c}_-}{n}$ y $n=n_++n_-$, sustituimos en la expresión anterior

$$\begin{aligned}
-\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'+n_-\beta'\mathbf{c}\mathbf{c}_-'-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'+n_+\beta'\mathbf{c}\mathbf{c}_+' &= n(\mathbf{c}_--\mathbf{c}_+)\\
-\beta'\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'+n_-\beta'\left(\frac{n_+\mathbf{c}_++n_-\mathbf{c}_-}{n}\right)\mathbf{c}_-'-\beta'\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'+n_+\beta'\left(\frac{n_+\mathbf{c}_++n_-\mathbf{c}_-}{n}\right)\mathbf{c}_+' &= n(\mathbf{c}_--\mathbf{c}_+)\\
\left(\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'-\frac{n_-n_+\mathbf{c}_+\mathbf{c}_-'-n_-n_-\mathbf{c}_-\mathbf{c}_-'}{n}+\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'-\frac{n_+n_+\mathbf{c}_+\mathbf{c}_+'-n_+n_-\mathbf{c}_-\mathbf{c}_+'}{n}\right)\beta &= n(\mathbf{c}_+-\mathbf{c}_-)\\
\left(\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'-\frac{n_-n_+\mathbf{c}_+\mathbf{c}_-'+n_-(n-n_+)\mathbf{c}_-\mathbf{c}_-'}{n}+\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'-\frac{n_+(n-n_-)\mathbf{c}_+\mathbf{c}_+'-n_+n_-\mathbf{c}_-\mathbf{c}_+'}{n}\right)\beta &= n(\mathbf{c}_+-\mathbf{c}_-)\\
\left(\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i'+\sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i'+-n_-\mathbf{c}_-\mathbf{c}_-'-n_+\mathbf{c}_+\mathbf{c}_+'+\frac{n_-n_+}{n}\left(-\mathbf{c}_+\mathbf{c}_-'+\mathbf{c}_-\mathbf{c}_-+\mathbf{c}_+\mathbf{c}_+'+-\mathbf{c}_-\mathbf{c}_+'\right)\right)\beta &= n(\mathbf{c}_+-\mathbf{c}_-)
\end{aligned}$$

Ocupemos ahora la **Propiedad 1,**

$$\left(\sum_{i:y_i=-1}\mathbf{x}_i\mathbf{x}_i' + \sum_{i:y_i=1}\mathbf{x}_i\mathbf{x}_i' + -n_-\mathbf{c}_-\mathbf{c}_-' - n_+\mathbf{c}_+\mathbf{c}_+' + \frac{n_-n_+}{n}\left(-\mathbf{c}_+\mathbf{c}_-' + \mathbf{c}_-\mathbf{c}_- + \mathbf{c}_+\mathbf{c}_+' + -\mathbf{c}_-\mathbf{c}_+'\right)\right)\beta = n(\mathbf{c}_+ - \mathbf{c}_-)$$

$$\left(\mathbf{S}_W + \frac{n_-n_+}{n}(\mathbf{c}_+ - \mathbf{c}_-)(\mathbf{c}_+ - \mathbf{c}_-)'\right)\beta = n(\mathbf{c}_+ - \mathbf{c}_-)$$

$$\left(\mathbf{S}_W + \frac{n_-n_+}{n}\mathbf{S}_B\right)\beta = n(\mathbf{c}_+ - \mathbf{c}_-) \blacksquare$$

Propiedad: 4. *Ocupando la* ***Propiedad 2****, se cumple que el mínimo*

$$\beta \sim \mathbf{S}_W^{-1}(\mathbf{c}_+ - \mathbf{c}_-).$$

Demostración de la Propiedad: 4. De la **Propiedad 2** tenemos que $\mathbf{S}_B\beta$ es un múltiplo del vector $(\mathbf{c}_+ - \mathbf{c}_-)$, llamemosla γ a este factor. Entonces, desarrollando el mínimo encontrado de la **Propiedad 3**, tenemos que

$$\begin{aligned}
\left(\mathbf{S}_W + \frac{n_-n_+}{n}\mathbf{S}_B\right) &= n(\mathbf{c}_+ - \mathbf{c}_-) \\
\mathbf{S}_W\beta + \frac{n_-n_+}{n}(\mathbf{c}_+ - \mathbf{c}_-)\gamma &= n(\mathbf{c}_+ - \mathbf{c}_-) \\
\mathbf{S}_W\beta &= -\frac{n_-n_+}{n}(\mathbf{c}_+ - \mathbf{c}_-)\gamma + n(\mathbf{c}_+ - \mathbf{c}_-) \\
\mathbf{S}_W\beta &= (\mathbf{c}_+ - \mathbf{c}_-)\left(-\frac{n_-n_+\gamma}{n} + n\right) \\
\beta &= \mathbf{S}_W^{-1}(\mathbf{c}_+ - \mathbf{c}_-)\left(-\frac{n_-n_+\gamma}{n} + n\right).
\end{aligned}$$

De lo anterior, podemos concluir que $\beta \sim \mathbf{S}_W^{-1}(\mathbf{c}_+ - \mathbf{c}_-)$ ■.

1.3. Ejemplo en 2D, de FDA usando mínimos cuadrados.

Con la ayuda de la librería sklearn (makeblobs) generamos una muestra aleatoria gaussianas isotrópicas para agrupación (ver Figura 1.1), consideramos una muestra de tamaño 300 con 2 categorías (-1: categoría azul y 1: categoría naranja). Y cada categoría tiene un tamaño de muestra de 150.

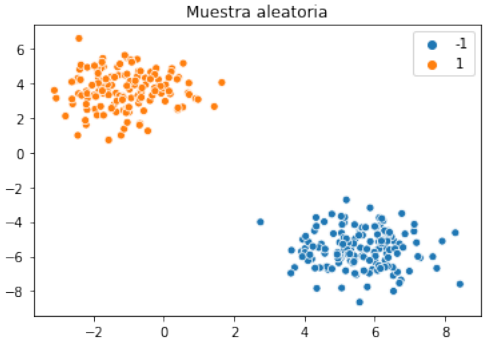

Figura 1.1: Dataset makeblobs

Entonces, tenemos que

$$\theta(1) = \frac{n}{n_+} = \frac{300}{150} = 2, \qquad \theta(-1) = -\frac{n}{n_-} = -\frac{300}{150} = -2.$$

Ahora, procedemos a utilizar la función LinearRegression() para obtener los estimadores por medio de mínimos cuadrados. En donde nuestro vector $Y = [\theta(y_1), \cdots, \theta(y_{300})]$ y la matriz X. Los estimadores obtenidos son:

$$\beta_0' = 0{,}76, \quad \beta = [-0{,}22274088, 0{,}26050276].$$

Observando los resultados del clasificador (Ver Figura 1.2), notamos que separa perfectamente las dos categorías de la muestra. Por lo que podemos concluir que es un buen clasificador.

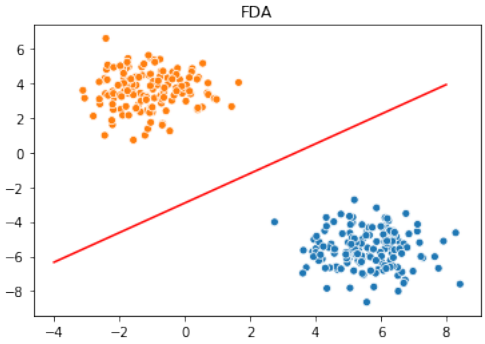

Figura 1.2: Clasificación binaria usando FDA

1.4. Ejemplo: Clasificación con datos atípicos

Nuestro conjunto de datos que utilizamos son la muestra del inciso anterior y generando 5 puntos más en un rango distinto a las dos categorías (datos atípicos). Primero utilizamos mínimos cuadrados para ajustar el clasificador como en la subsección anterior (ver Figura 1.3). Podemos notar que nuestro clasificador esta muy afectado por nuestros datos atípicos, es decir, podemos notar que FDA no es robusto a datos atípicos.

Para hacer más robusto el clasificador FDA consideramos usar mínimos cuadrados ponderados, es decir, nuestra función de costo a minimizar es

$$\sum_{i=1}^{n} w_i \left(\theta(y_i) - \beta' \mathbf{x}_i - \beta_0\right)^2.$$

Donde w_i es un factor de importancia del punto i. Nosotros consideramos este factor es inversamente proporcional a la distancia (euclideana) del punto a su centroide de la categoría a la cual pertenece. Es decir,

$$\sqrt{w_i} = \frac{1}{dist(\mathbf{x}_i, c_i)}, \qquad c_i \text{ es el centrioide de la categoría de } i.$$

Entonces, para hacer más robustos FDA tenemos que trabajar con los datos: $\mathbf{Y}_{new} = [\sqrt{w_1}\theta(y_1), \cdots, \sqrt{w_{305}}\theta(y_{305})]$ y $X = \sqrt{w} * X$. Y los estimadores de mínimos cuadrados se implemente de la misma manera que

en la subsección anterior.

Comparando ambas aplicaciones (ver Figura 1.3), **podemos observar que con la implementación de mínimos cuadrados ponderados el clasificador binario FDA se vuelve más robusto.**

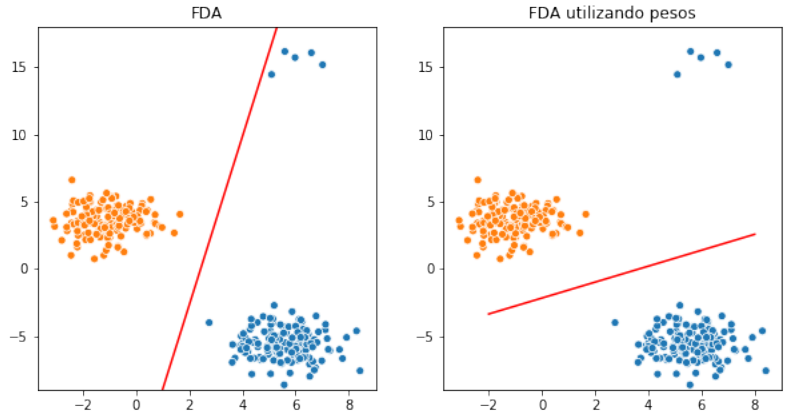

Figura 1.3: Comparación FDA controlando los datos robustos.

2. Clasificación de MNIST (imágenes de dígitos)

2.1. Introducción

Consideremos los datos MNIST de dígitos escritos a mano de 28 ×28 píxeles. El objetivo es poder clasificar las 70000 imágenes en los dígitos que aparecen en ellas.

2.2. Implementación de los distintos métodos clasificación

Primeramente partimos nuestro conjunto de datos (imágenes) en dos conjuntos: entrenamiento y prueba. En el conjunto de entrenamiento tenemos 60000 datos y para no tener problemas con una muestra desbalanceada, intentamos en tener una proporción de los dígitos muy parecidas (Ver Cuadro 2.1). Lo anterior también se considero para el conjunto de datos de prueba pero en este caso tenemos 10000 datos. **Debido al tiempo de ejecución de algunos algoritmos utilizados no se considero usar validación cruzada para los ajustes**, pero es un punto muy importante a controlar.

Dígito	Conteo	Dígito	Conteo
0	5923	5	5421
1	6742	6	5918
2	5958	7	6265
3	6131	8	5851
4	5842	9	5949

Cuadro 2.1: Conteos de los dígitos en los datos de entrenamiento.

Todos los métodos se implementaron en python, usando la librería de sklearn ([1]).

2.2.1. Métricas de Evaluación

Para este ejercicio podríamos considerar solo el **accuracy** para determinar que tan bueno es el ajuste, ya que se esta utilizando una muestra balanceada. Además de que en este ejercicio no es crucial controlar algún tipo de error especifíco (falsos positivos o verdaderos negativos), como en otros ejercicios por ejemplo referentes a estudios humanos.

Pero para tener una mejor perspectiva de los errores consideraremos también las siguientes métricas: **precisión, recall y F1 score**.

2.2.2. Baseline

Consideraremos como *baseline* un método de regresión multivariada, es decir, tenemos

$$\mathbf{Y} = \mathbf{X}\hat{B},$$

donde $\mathbf{Y}_{n\times|K|}$ es una matriz indicadora, donde cada renglón tiene ceros excepto en lugar que corresponde el valor y_k, donde colacamos un 1. $\mathbf{X}_{n\times 784}$ es la matriz de características y $\hat{B}$ es la matriz cuyas columnas contienen los $|K|$ coeficientes correspondientes $\hat{\beta}_k$. Con esta formulación, asumimos un modelo lineal para cada respuesta y_k :

$$\hat{\mathbf{y}}_k = \mathbf{X}\hat{\beta}_k,$$

y la clasificación para alguna observación $\mathbf{x}$ se obtiene mediante

$$\hat{C}(\mathbf{x}) = \arg\max_{k\in K} \hat{y}_k.$$

2.2.3. Resultados Baseline

Con la sección anterior, implementamos el método en python al conjunto de entrenamiento y clasificamos las imágenes. Posteriormente calculamos los métricas de error en el conjunto de entrenamiento y prueba, en de entrenamiento para observar si no existe algún sobreajuste en los datos y los errores del conjunto de prueba para contrastar el rendimiento de este modelo con otros modelos (Ver Cuadro 2.2).
Comparando el accuracy, precision, recall y f1−score de ambos conjunto de datos podemos observar que son muy parecidos, **por lo que podemos decir que no existe un sobreajuste en los datos**. Concentrándonos en las métricas del conjunto de prueba, podemos observar que en general todos los errores están por arriba de 0.80 a excepción del recall del digíto 2 y 5 (ver Figura 2.1). Es decir, en estos dos dígitos nos estamos equivocando en clasificando otros números en 2 y 5. Si observamos **los f1−score y el accuracy podemos notar que todos están por arriba del 0.80, por lo que podemos decir que este modelo tiene un buen rendimiento.**

Cuadro 2.2: Métricas de los conjuntos de entrenamiento y prueba.

Cuadro 2.3: Métricas de entrenamiento

dígito	precision	recall	f1-score
0	0.90	0.96	0.93
1	0.82	0.97	0.89
2	0.91	0.81	0.85
3	0.84	0.84	0.84
4	0.84	0.89	0.86
5	0.87	0.74	0.80
6	0.89	0.93	0.91
7	0.87	0.87	0.87
8	0.84	0.75	0.80
9	0.81	0.80	0.81
accuracy			0.86

Cuadro 2.4: Métricas de prueba

dígito	precision	recall	f1-score
0	0.91	0.96	0.93
1	0.83	0.98	0.90
2	0.92	0.79	0.85
3	0.85	0.87	0.86
4	0.81	0.90	0.85
5	0.88	0.74	0.80
6	0.88	0.91	0.90
7	0.85	0.86	0.86
8	0.84	0.78	0.81
9	0.84	0.79	0.82
accuracy			0.86

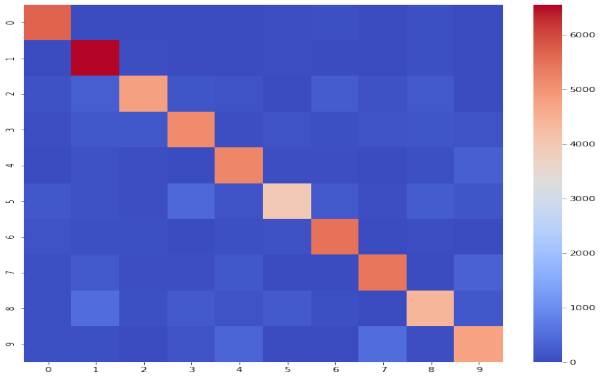

Figura 2.1: Matriz de confusión del conjunto de prueba.

Con las métricas anteriores, intentaremos mejóralas utilizando otros métodos de clasificación: Análisis discriminante lineal (LDA), Análisis discriminante cuadrática (QDA) y Regresión logística.

2.2.4. LDA, QDA y Regresión logística

Para este primer acercamiento no consideraremos ninguna transformación o representación de los datos, es decir, simplemente usamos los datos originales.
Primero ocupamos el **método LDA** y calculamos las métricas de error (ver Cuadro 2.5 y Figura 2.2). Comparando el accuracy de los dos conjuntos observamos que son muy parecidos, por lo que podemos decir que no existe un sobreajuste. Ahora si comparamos las métricas del conjunto de entrenamiento de este método contra el método *baseline*, observamos una pequeña mejoría en casi todas las métricas, a excepción de la presición de los dígitos 3, 4, 5 y el recall en los dígitos 1 y 6. Y si comparamos los f1−score, en los dígitos 8 y 9 en LDA son menores y en los dígitos restantes son mayores. **En conclusión, este modelo tiene un mejor rendimiento en comparación con el** *baseline*, **pero esta diferencia es muy pequeña.**

Cuadro 2.5: Métricas de los conjuntos de entrenamiento y prueba, usando LDA.

Cuadro 2.6: Métricas de entrenamiento

dígito	precision	recall	f1-score
0	0.95	0.94	0.95
1	0.87	0.96	0.91
2	0.92	0.82	0.86
3	0.86	0.85	0.86
4	0.86	0.90	0.88
5	0.83	0.82	0.83
6	0.93	0.92	0.92
7	0.92	0.84	0.88
8	0.81	0.80	0.80
9	0.78	0.86	0.82
accuracy			0.87

Cuadro 2.7: Métricas de prueba

dígito	precision	recall	f1-score
0	0.94	0.96	0.95
1	0.89	0.97	0.93
2	0.92	0.79	0.85
3	0.87	0.87	0.87
4	0.84	0.90	0.87
5	0.84	0.82	0.83
6	0.91	0.89	0.90
7	0.91	0.84	0.88
8	0.80	0.81	0.80
9	0.81	0.85	0.83
accuracy			0.87

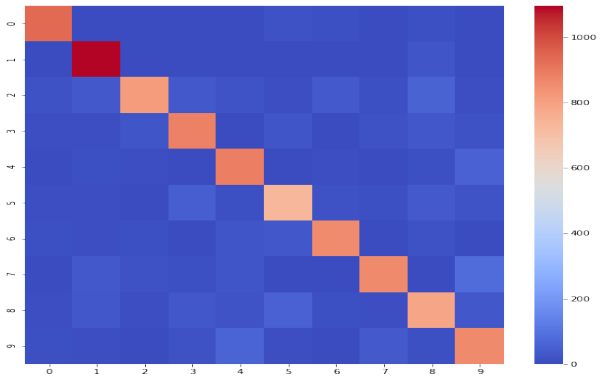

Figura 2.2: Matriz de confusión del conjunto de prueba, usando LDA.

Ahora continuemos ocupando **el método QDA**, realizamos analogamente el análisis anterior. Comparando el accuracy de los dos conjuntos observamos que son muy parecidos, por lo que podemos decir que no existe un sobreajuste. **Para este modelo no tiene mucho sentido realizar las comparaciones uno a uno, ya que en la mayoría (o todas) las metrícas son peores en este método** (ver Cuadro 2.8). La razón principal por la cuál no es muy bueno este enfoque es que tal vez las variables que estamos utilizando sean colineales, lo que provoca que la matriz de covarianzas no pueda ser invertible y no converga. Entonces para este método es conveniente reducir el número de variables con PCA o algún otro método.

Cuadro 2.8: Métricas de los conjuntos de entrenamiento y prueba, usando QDA.

Cuadro 2.9: Métricas de entrenamiento

dígito	precision	recall	f1−score
0	0.37	0.98	0.54
1	0.86	0.96	0.90
2	0.94	0.27	0.42
3	0.70	0.39	0.51
4	0.98	0.19	0.32
5	0.96	0.17	0.29
6	0.70	0.98	0.82
7	0.94	0.44	0.60
8	0.53	0.63	0.58
9	0.50	0.96	0.66
accuracy			0.61

Cuadro 2.10: Métricas de prueba

dígito	precision	recall	f1−score
0	0.34	0.96	0.51
1	0.89	0.95	0.92
2	0.89	0.21	0.34
3	0.63	0.35	0.45
4	0.95	0.16	0.27
5	0.89	0.15	0.26
6	0.68	0.96	0.79
7	0.92	0.39	0.55
8	0.50	0.62	0.56
9	0.49	0.93	0.64
accuracy			0.58

Y por último, utilizaremos **Regresión Logística** para ello realicemos el mismo análisis que los anteriores. En este caso consideraremos dos valores para el parámetro *multi_class*: *ovr* y *multinomial*. Usando *ovr*, entonces se ajusta un problema binario para cada etiqueta y para *multinomial*, la pérdida minimizada es el ajuste de pérdida multinomial en toda la distribución de probabilidad, incluso cuando los datos son binarios ([1]).

Los métricas con ambos valores del parámetro se observan en el Cuadro 2.11 y Figura 2.3. Primero observemos que las diferencias entre los errores del conjunto de datos de entrenamiento y prueba son muy parecidos, por lo que suponemos que no existe sobreajuste. Ahora, considerando las métricas del conjunto de prueba observamos que considerando cualquier valor del parámetro existen pequeñas diferencian en las métricas, pero en general son muy parecidos. **Si comparamos estas métricas con el método** *baseline* **observamos que todas las métricas son mejores,** de hecho en este caso los f1−score todos son mayores a 0.89 y el accuary es de 0.93. **Por lo que, concluimos que este método fue mejor que todos los anteriores (considerando los datos originales, es decir, sin transformaciones ni otra representación).**

Y observando la matriz de confusión, podemos notar que el clasificador es muy buen, y que el dígito que más tiene problemas en clasificar es el número 8.

Cuadro 2.11: Métricas de los conjuntos de entrenamiento y prueba, usando Regresión Logistica.

Cuadro 2.12: Métricas de entrenamiento

dígito	precision	recall	f1-score
0	0.97−0.96	0.98−0.98	0.97−0.97
1	0.97−0.96	0.98−0.98	0.97−0.97
2	0.94−0.93	0.92−0.91	0.93−0.92
3	0.92−0.91	0.92−0.90	0.92−0.90
4	0.95−0.93	0.95−0.94	0.95−0.93
5	0.92−0.91	0.90−0.87	0.91−0.89
6	0.96−0.95	0.97−0.96	0.96−0.96
7	0.95−0.94	0.95−0.94	0.95−0.94
8	0.90−0.86	0.91−0.88	0.90−0.87
9	0.92−0.90	0.92−0.90	0.92−0.90
accu			0.94-0.93

Cuadro 2.13: Métricas de prueba

dígito	precision	recall	f1−score
0	0.95−0.96	0.98−0.98	0.96−0.97
1	0.96−0.96	0.98−0.98	0.97−0.97
2	0.93−0.93	0.90−0.91	0.91−0.92
3	0.90−0.91	0.92−0.90	0.91−0.90
4	0.94−0.93	0.93−0.94	0.94−0.93
5	0.91−0.91	0.86−0.87	0.88−0.89
6	0.94−0.95	0.95−0.96	0.95−0.96
7	0.93−0.94	0.92−0.94	0.93−0.94
8	0.87−0.86	0.89−0.88	0.88−0.87
9	0.91−0.90	0.92−0.90	0.91−0.90
accu			0.93-0.93

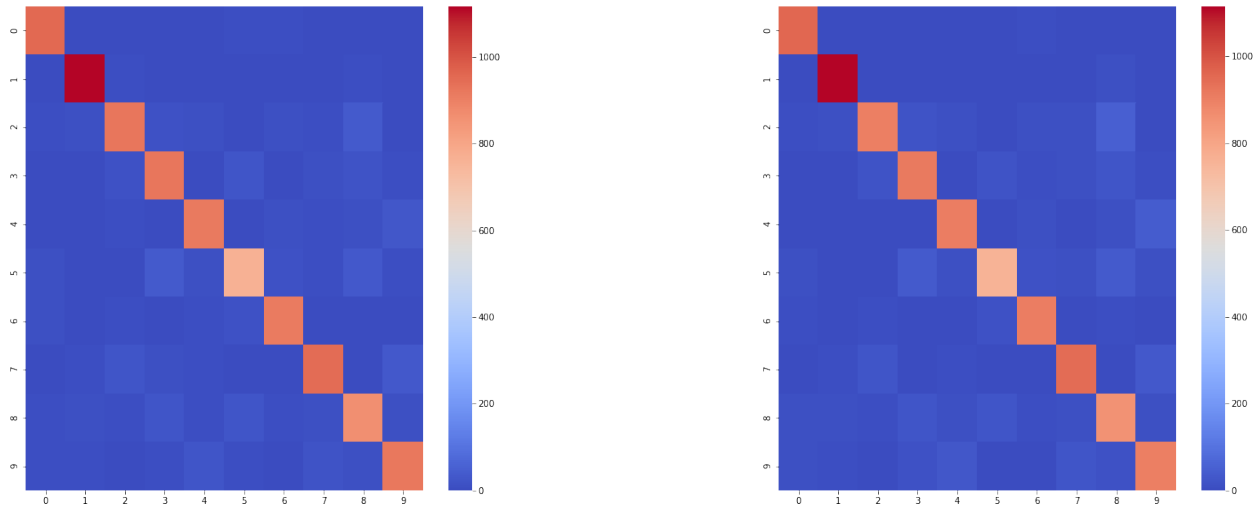

Figura 2.3: Matriz de confusión del conjunto de prueba, usando Reg Log.

2.3. Nueva representación de los datos.

Para este caso no se considera una estandarización o escalamiento de los datos, debido a que los métodos LDA, QDA y Regresión Logística son robustos a estas transformaciones.

El siguiente paso a considerar fue reducir la dimensionalidad. **Debido a que tenemos alrededor de 800 variables, posiblemente existan variables que no sean relevantes (exista una alta correlación). Por lo cual, consideramos utilizar PCA para reducir el número de variables.** Los métodos LDA y Regresión Logística tuvieron pequeños cambios, **pero QDA** (que si recordamos era el método que presentaba los peores errores de este ejercicio) **presento un cambio muy significativo, al grado de ser el método con las mejores métricas.**

2.3.1. Comparación de métricas: QDA y Regresión Logística

Primeramente **utilizamos PCA para reducir el número de variables a 200 de nuestro conjunto de datos de entrenamiento** (no probé con distintos valores de número de componentes). Posteriormente proyectamos nuestro conjunto de prueba con los valores del PCA ajustado. Y finalmente **utilizamos QDA para clasificar las imágenes**.

Comparando con las métricas de QDA y las que obtuvimos anteriormente de los modelos de Regresión Logística, los f1−score en la mayoría de los casos están por arriba o igual que el mejor modelo de Regresión Logística (ver Cuadro 2.14 y Figura 2.4). El dígito 8 es el que presenta más error de clasificación, **pero en general podrías concluir que este ajuste es el mejor que todos los que se probaron.**

Cuadro 2.14: Comparación de metricas en el conjunto de entrenamiento: Reg Log y QDA.

Cuadro 2.15: Métricas usando Reg Log

dígito	precision	recall	f1−score
0	0.95	0.98	0.96
1	0.96	0.98	0.97
2	0.93	0.90	0.91
3	0.90	0.92	0.91
4	0.94	0.93	0.94
5	0.91	0.86	0.88
6	0.94	0.95	0.95
7	0.93	0.92	0.93
8	0.87	0.89	0.88
9	0.91	0.92	0.91
accuracy			0.93

Cuadro 2.16: Métricas usando QDA

dígito	precision	recall	f1−score
0	0.97	0.98	0.98
1	0.99	0.95	0.97
2	0.93	0.96	0.94
3	0.94	0.92	0.93
4	0.96	0.96	0.96
5	0.95	0.91	0.93
6	0.97	0.95	0.96
7	0.97	0.93	0.95
8	0.83	0.95	0.88
9	0.94	0.93	0.93
accuracy			0.94

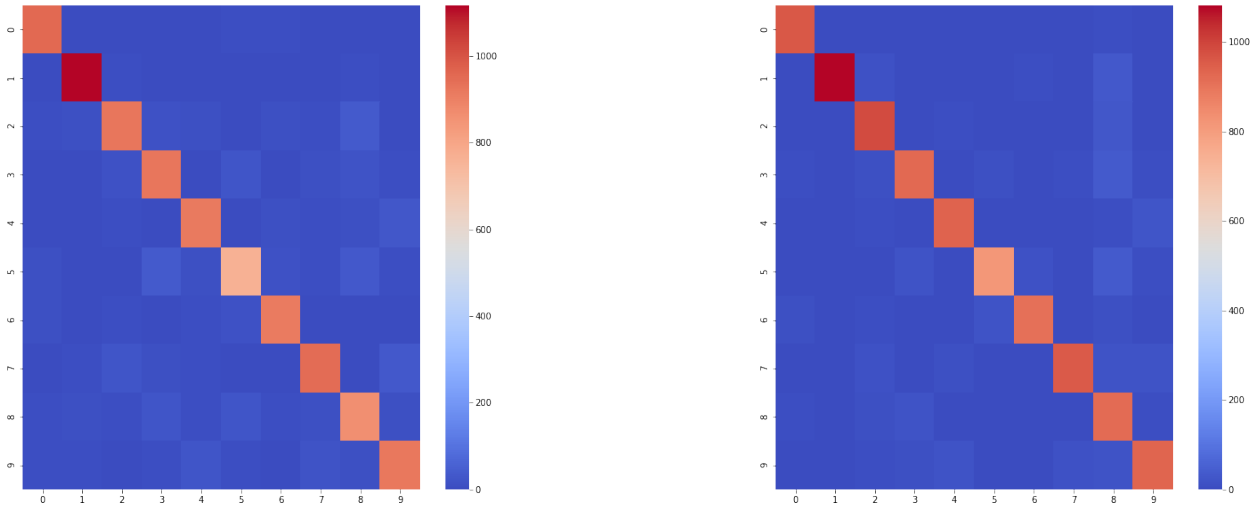

Figura 2.4: Matriz de confusión del conjunto de prueba, usando Reg Log y QDA

Podemos concluir, que tenemos que tener cuidado con los métodos QDA y LDA ya que estos tienen

supuestos que deben de cumplir para tener un mejor rendimiento. Claramente se pudo observar este efecto en este ejercicio.

3. Análisis de texto y de sentimientos.

Tenemos 400 opiniones de usuarios tanto positivas como negativas acerca de los siguientes productos: automóviles, hoteles, lavadoras, libros, teléfonos celulares, música, computadoras y películas.

3.1. Bad of Words (BOW)

Antes a realizar algún análisis con los datos, tenemos que realizar un preproceso y normalización del texto. Nosotros realizamos lo siguiente:

1. Convertimos todas las letras a minúsculas.
2. Eliminamos signos de puntuación, acentos y otros signos diacríticos.
3. Eliminamos caracteres repetidos (ejemplo: taaaaanto), saltos de línea y espacios en blanco.
4. Eliminamos palabra funcionales (stop words).
5. Y por último, aplicamos stemming y lematización.

Lo anterior se realizo debido a que las opiniones estaban muy , lo que provocaría un sesgo. Se probó con diferentes combinaciones, pero el conjunto de datos que presentaba mejores resultados fue cuando se realizaron las anteriores tareas de limpieza.

Además, consideremos diferentes valores para tamaño de vocabulario. Nos dimos cuenta que entre más pequeño sea el tamaño del vocabulario existe una mayor cantidad de opiniones con distancias cercanas a 1, en cambio cuando el tamaño era demasiado grande las distancias eran muy cercanas a 0. Es decir, lo anterior tiene mucho sentido debido a que la distancia esta muy relacionada con las palabras que se usaran en el vocabulario. Seleccionamos el tamaño de vocabulario observando la matriz de distancias, **ya que con un valor de $V = 3000$ es sencillo identificar de forma visual los grupos de opiniones de productos y sentimientos.**

Y por último, otro factor a considerar es el **rango de n−valores para diferentes n−gramas de palabras.** Probamos con diferentes rangos, pero si el rango era grande la interpretación de las distancias era complicada. Y si el valor era grande, la información era mínima. **Por lo cuál solo consideremos una palabra, es decir, el rango fue de** $(1, 1)$**.**

3.2. Matriz de similaridades usando BOW y la distancia coseno.

Con la representación BOW descrito en la sección anterior, procedemos a calcular la matriz de similaridades considerando la distancia coseno:

$$k(x, y) = \frac{xy^T}{||x||||y||}.$$

Si ordenamos las opiniones de por el número de artículos, y comparando la matriz de distancias de las opiniones con una matriz de distancias realizadas solo con la palabra del artículo a que opinan (ver Figura 3.3), es decir, queremos contrastar si podemos reconocer las opiniones según los

Figura 3.1: Matriz de distancias de las opiniones.

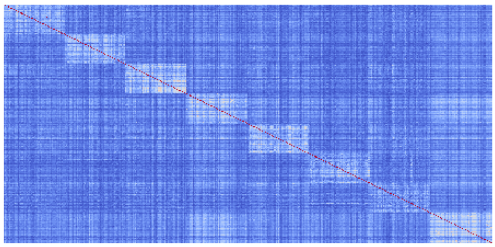

Figura 3.2: Matriz de distancias del vector de productos.

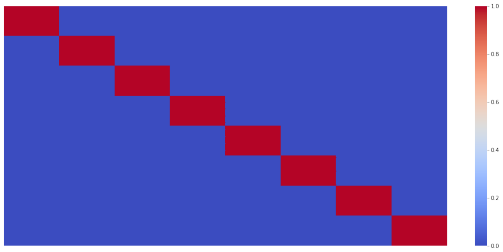

Figura 3.3: Reconocimiento visual de patrones de los productos en las opiniones.

productos calificados. Comparando estas dos matrices de distancias, podemos observar claramente el patrón de los productos en las opiniones.
Ahora realicemos un nuevo ordenamiento para comparar la matriz de distancias de las opiniones y con una matriz de considerando un vector compuestas de 400 registros cada uno con los productos y sentimiento (ver Figura 3.6). Para este caso, **se observa un poco el patrón de los segmentos de los productos y sentimientos de las opiniones, claramente no es para todas las combinaciones pero si es un buen proxy de segmentación.**

Figura 3.4: Matriz de distancias de las opiniones.

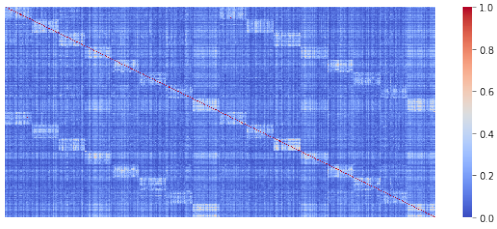

Figura 3.5: Matriz de distancias del vector de productos y sentimientos..

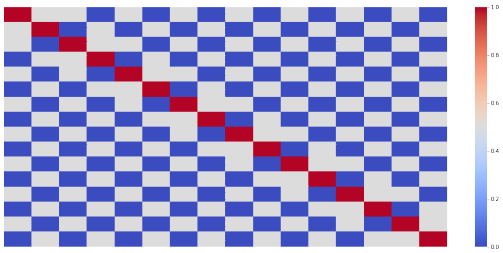

Figura 3.6: Reconocimiento visual de patrones de los productos y sentimientos en las opiniones.

3.3. Clustering en las representaciones BOW.

Usando la matriz de similaridades calculada en el sección anterior, procedemos a implementar distintos métodos de clustering (KMeans, Kernel KMeasn, Spectral Clustering) para encontrar las opiniones a partir de los productos que evaluaban y el sentimiento de la opinión (positiva o negativa).

Para presentar **los resultados procedimos primero representar la representación BOW en dos dimensiones, con el objetivo de observar visualmente los clustering.** Probamos distintos métodos de reducción de dimensionalidad (PCA, KPCA, TSN−E y SE). **La mejor representación fue considerando Spectral Embedding (**$gamma = 0{,}6$ **y** $n\_neigbors = 10$**)** (ver Figura 3.7).
Debido a que tenemos las categorías de las opiniones podemos validar de forma cuantitativa los clusters encontrados. En la vida real cuando clustering no supervisado esto no es

posible, pero para fundamentar mejor las visualizaciones de este análisis utilizamos las métricas de para un ajuste de clasificación.

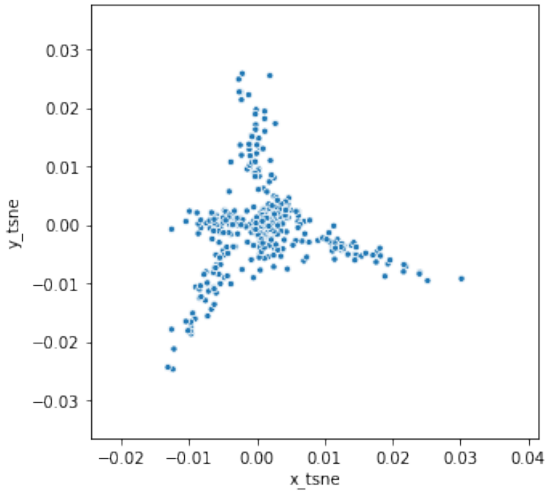

Figura 3.7: Representación 2D usando SE

El mejor modelo que encuentra algunos patrones interesantes fue utilizando Spectral Embedding Clustering. Primero intentamos encontrar las opiniones de acuerdo a los sentimientos, para ello consideramos buscar dos clustering en nuestro método (ver Figura 3.8). En este caso no pudimos encontrar las categorías esperadas de los sentimientos (*yes*, *no*), de hecho si comparamos con las clasificaciónes reales el accuracy final sería de 0.52. Lo cual nos puede indicar que nuestros clusters no es buena para encontrar los sentimientos.

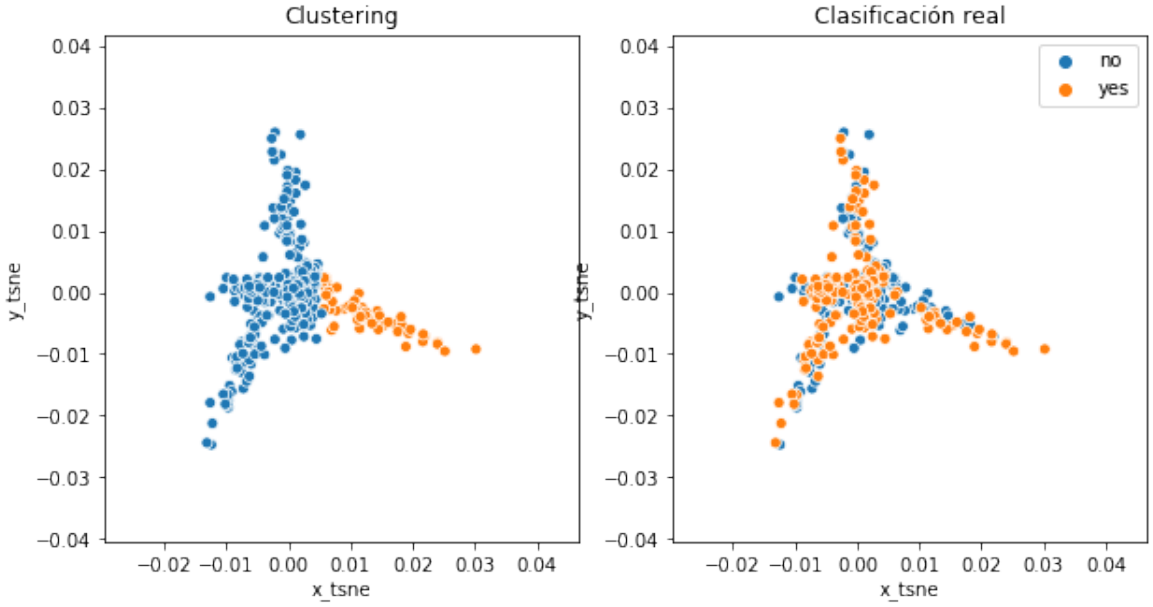

Figura 3.8: Clustering para los sentimientos

Ahora, si buscamos 8 categorías en nuestro conjunto de datos obtenemos los 8 clusters que se ven

en la Figura 3.9. Si comparamos los clustering encontrados con las categorías reales obtenemos un accuracy del 0.94, además de que las métricas de precisión, recall y f1−score son buenas. Es decir, **este modelo pudo identificar perfectamente las categorías de los productos en las opiniones**.

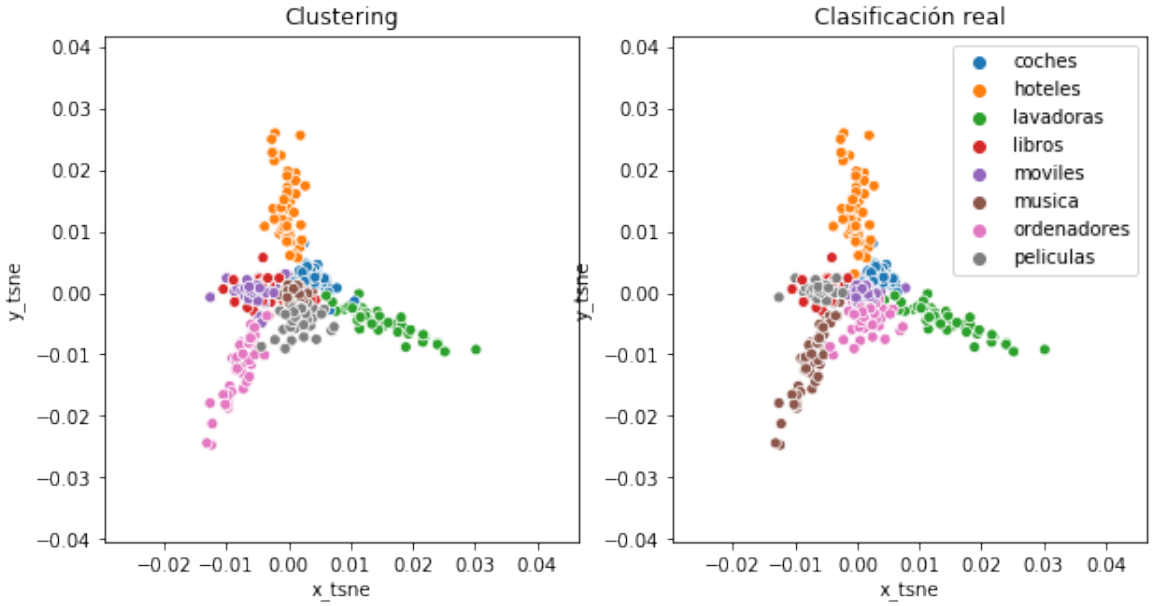

Figura 3.9: Clustering para los productos

Por último, intentamos encontrar las opiniones para cada producto y por sentimiento. En total son 16 categorías diferentes, pero en general no pudimos encontrar todas estas categorías.

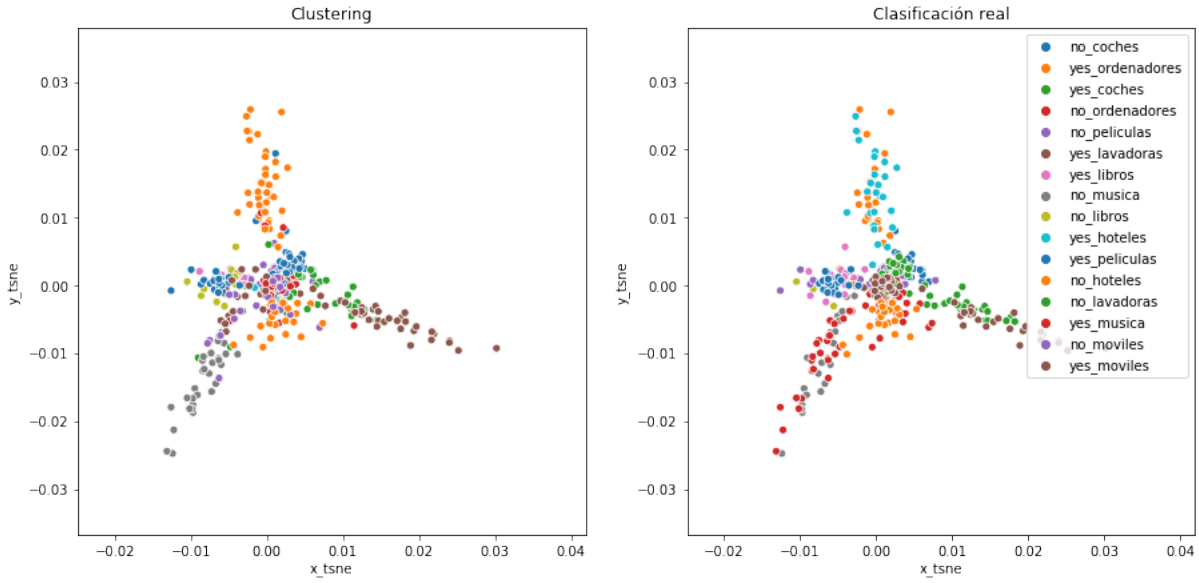

Figura 3.10: Clustering para los sentimientos y productos

Algo interesante a observar es que hay ciertas categorías de productos y sentimientos que si se pueden

identificar muy bien (lavadoras con el sentimiento *yes*, o hoteles con los sentimientos *no*), hay otras combinaciones que en las cuales se confunden con los sentimientos. Por ejemplo, las opiniones que hablan de los coches no es posible separar los sentimientos: *yes y no*.

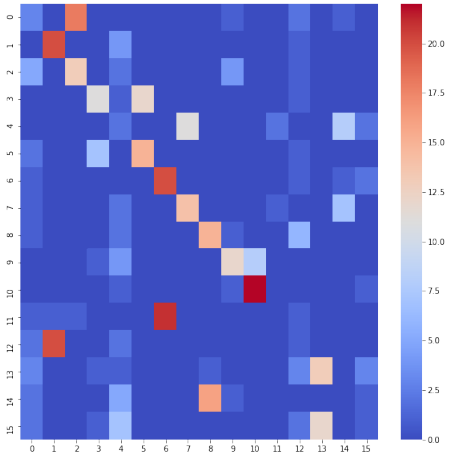

Figura 3.11: Matriz de confusión de las opiniones considerando los sentimientos y productos

Por lo anterior, **podemos concluir que es posible identificar las opiniones por productos pero no es posible identificar por sus sentimientos. Y cuando se intentar identificar las opiniones por productos y sentimientos, la mayoría de las combinaciones no es posible identificarlas ya que se confunden con sus respectivas opiniones pero con distinto sentimientos.**

3.4. Clasificación supervisada de las opiniones respecto a los sentimientos y productos.

El objetivo de esta subsección fue poder clasificar las opiniones conforme a los productos y sentimientos, mediante un modelo supervisado.

Para ello primero partimos nuestro conjunto de datos en dos conjuntos $(70 - 30)$: entrenamiento y prueba. **En el conjunto de entrenamiento consideramos varios modelos con diferentes valores en sus parámetros y además diferentes representación de los datos**. Los modelos ocupados fueron **LDA, QDA y Regresión logística, las representaciones de los datos consideradas fue la reducción de dimensionalidad de las variables usando PCA donde probamos diferentes número de componentes principales**. Y por último, en el modelo de regresión logística utilizamos **regularización** $l2$ debido a que teníamos un problema de sobreajuste, consideramos diferentes valores de regularización: 0.1, 0.05, 0.0094, 0.005. Para el modelo de regresión logística ocupamos **validación cruzada (cv=9)** para controlar el sobreajuste de los datos, solo se realizo en este modelo debido a que en la librería de sklearn ya viene una implementación de este modelo usando CV.

3.4.1. Clasificación de los sentimientos

Empecemos presentando los resultados de los modelos para poder predecir el sentimiento a partir de los sentimientos. Los modelos **LDA y QDA** tienen un muy buen rendimiento cuando se usan

una mayor cantidad de componentes principales en el conjunto de entrenamiento. P**ero el principal problema de estos modelos fue que sobreajustaron** a los datos, ya que al calcular las métricas de error en el conjunto de prueba estas eran muy malas. Por ejemplo, **usando LDA con 200 componentes principales el accuracy en el conjunto de entrenamiento es de 0.96. En cambio el accuracy en el conjunto de prueba es de 0.58, lo que implica que hay un sobreajuste en los datos.**

Ahora, **utilizando Regresión logística la mejor combinación fue ocupando una penalización de 0.005 y 200 componentes principales** (ver Cuadro 3.1). Cabe mencionar que existe una gran diferencia en las métricas del conjunto de entrenamiento y prueba, pero como se utilizo validación cruzada para estos resultados se cree que están mejor fundamentados.

Cuadro 3.1: Comparación de métricas en el conjunto de entrenamiento y prueba, Reg Log.

Cuadro 3.2: Conjunto de entrenamiento

review	precisión	recall	f1−score
yes	0.99	0.98	0.99
no	0.98	0.99	0.98
accuracy			0.99

Cuadro 3.3: Conjunto de prueba

review	precisión	recall	f1−score
yes	0.74	0.65	0.69
no	0.69	0.77	0.72
accuracy			0.71

3.4.2. Clasificación de los productos

Para predecir de que producto están hablando a partir de las opiniones realizamos lo mismo que en las subsecciones anteriores. Para este caso, los modelos probados de QDA fueron los que presentaron peores métricas, debido a que se sobreajustaban bastante al conjunto de entrenamiento. Esto puede deberse a que este modelo no es robusto a valores extremos, lo que provoca un mal rendimiento en el conjunto de prueba. De igual manera se observa con los modelos de LDA, pero este hecho es menos notorio.

El mejor modelo es considerar ocupar regresión logística ocupando una regularización de 0.01 en los primeros 200 componentes principales (ver Cuadro 3.4). Las métricas la mayoría están muy cercanas a 1, lo que pudiera indicar un problema de sobreajuste pero debido a que se utilizo validación cruzada descartamos esta opción.

Cuadro 3.4: Comparación de métricas en el conjunto de entrenamiento y prueba, Reg Log.

Cuadro 3.5: Conjunto de entrenamiento

producto	precisión	recall	f1−score
coches	1.00	1.00	1.00
hoteles	1.00	1.00	1.00
lavadoras	1.00	1.00	1.00
libros	1.00	1.00	1.00
moviles	1.00	0.97	0.98
musica	0.97	1.00	0.98
ordenadores	0.97	1.00	0.98
peliculas	1.00	0.97	0.99
accuracy			0.99

Cuadro 3.6: Conjunto de prueba

review	precisión	recall	f1−score
coches	1.00	1.00	1.00
hoteles	1.00	1.00	1.00
lavadoras	1.00	1.00	1.00
libros	1.00	0.91	0.95
moviles	1.00	0.94	0.97
musica	1.00	1.00	1.00
ordenadores	0.95	1.00	0.97
peliculas	0.92	1.00	0.96
accuracy			0.98

3.5. Conclusión

Por un lado **podemos concluir que fue sencillo encontrar y predecir los patrones de los productos en las opiniones. Pero poder identificar si las opiniones eran positivas o negativas resulto un poco más complicado**, tal vez se deba a la limpieza del texto. Estaría interesante ver el efecto de una representación de BOW solo usando palabras que nos den indicios de disgusto o agradecimiento, es decir, considerar palabras muy especificas: gracias, excelente, pésimo, maravilloso, etc.

Y por otro lado, nos dimos cuenta de lo sensible que son los modelos. Es decir, realizar las tareas de validación cruzada, separar nuestro conjunto de entrenamiento y prueba nos ayuda a no cometer errores de sobreajuste. Además, los modelos de **LDA y QDA** son muy sensibles a valores extremos lo que hace que los errores aumenten en un conjunto de datos distinto al de entrenamiento.

4. Anexos

Todos los códigos utilizados para estos resultados se pueden encontrar en mi página personal de Gitgub: Enriquesec. En el repositorio Ciencia de Datos/Tareas/Tareas5/.

Referencias

[1] F. Pedregosa y col. “Scikit-learn: Machine Learning in Python”. En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.