Introducción a la clasificación

UCO ONLINE

Introducción a la clasificación

Introducción

A diferencia del problema de regresión, donde la variable objetivo es continua, aquellos casos donde se trata de predecir una variable de salida categórica se les denomina problemas de clasificación.

Consideremos un conjunto de datos (también llamado *dataset*) de *n* instancias. Cada instancia está compuesta por un conjunto de *m* atributos de entrada y una variable de salida. El objetivo de un método de clasificación es aprender un modelo a partir de dichos datos, que, dada una instancia para la cual no se conoce su clase, sea capaz de asignarle una.

Formalmente, el objetivo principal de los clasificadores se define como: *Dado un conjunto de entrenamiento S, con un conjunto de atributos* $X = \{x_1, x_2, \dots x_m\}$ *y una variable objetivo y de tipo nominal y a partir de una distribución desconocida D sobre el espacio de datos etiquetados, el objetivo es inducir un clasificador óptimo* Ω *con el menor error de generalización.*

El algoritmo de clasificación examinará los datos proporcionados de entrada, de los cuales "aprenderá" qué combinaciones de variables se asocian con cada valor de la clase objetivo. El aprendizaje no es más que la capacidad de inferir la definición general de los objetos de una clase dado un conjunto de ejemplos. Una vez el modelo conoce dichas relaciones, podrá predecir y asignar una clase a un nuevo registro que no tiene dicha variable de clase asociada.

Los atributos (también llamados campos, variables, o características) se reducen típicamente a dos tipos: nominal y numérico. Los atributos nominales pueden tomar valores de entre un conjunto de valores predefinido, y no puede calcularse distancia u orden entre los valores. En cambio, los atributos numéricos pueden tomar valores continuos (ya sean flotantes o enteros).

Usualmente, los datos utilizados para entrenar un clasificador tienen una estructura similar a la que se muestra en la Tabla 1. Como se observa, este conjunto de datos tiene *n* instancias, cada una compuesta de *m* atributos y un valor de clase. Como en este caso, un mismo conjunto de datos puede contener atributos tanto nominales como numéricos, mientras que la clase es siempre de tipo nominal.

Tabla 1. Ejemplo de dataset para clasificación.

	x_1	x_2	x_3	…	x_m	y
i_1	1.5	Va1	3		Vm1	C1
i_2	1.3	Va1	2		Vm2	C2
…			…			…
i_n	0.9	Va4	3		Vm1	C1

En cuanto a los posibles valores del atributo de clase, cabe destacar dos escenarios:

- Si el atributo de clase toma únicamente dos valores (generalmente descritos como clases positiva y negativa), se trata de clasificación binaria.
- Si el atributo de clase puede tomar más de dos valores, se habla de clasificación multi-clase. Igualmente, cada instancia estará asociada con una de esas clases previamente predefinidas, pero la elección será de entre 3 o más clases.

Cabe destacar que la salida final del clasificador será siempre la clase con la que se asocia el patrón o instancia de entrada. Sin embargo, en algunos casos los clasificadores podrían proporcionar como salida no un valor nominal sino numérico, indicando cuán propensa es la instancia en cuestión de pertenecer a una clase en particular. Posteriormente, se escogería como clase asociada aquella que tenga un valor mayor de importancia, o crear un orden o crear un orden entre ellas si fuera necesario.

Modelos lineales

En este sección se presentan algunos de los modelos de clasificación más clásicos, los modelos lineales.

Clasificador lineal clásico

Uno de los clasificadores más simples que pueden construirse son los llamados clasificadores lineales. En un clasificador lineal, la salida es simplemente la suma de cada uno de los atributos por su peso asociado (por tanto, todos los atributos deben ser de tipo numérico). La dificultad en el entrenamiento del modelo viene a la hora de buscar los pesos que hagan que el modelo obtenga la salida deseada a partir de los valores de entrada. El proceso es similar al realizado en regresión; sin embargo, en lugar de buscar una función que se ajuste a los puntos existentes, se busca una función que separe de la mejor manera posible los datos disponibles.

De modo general, los clasificadores lineales se utilizan para resolver problemas de clasificación binaria, ya que una línea (o plano, o hiperplano) permite separar dos clases. Dicha separación se llama comúnmente frontera de decisión. En la Figura 1 se muestra un ejemplo donde se observan las instancias de datos de dos clases distintas (utilizando dos atributos) y la frontera de decisión que las separa. Los puntos de la línea que separa los datos vienen dados por la ecuación $2.0 - 0.5 * PetalLength - 0.8 * PetalWidth = 0$. Dada una instancia de test, se le pasa a dicha ecuación los valores de los atributos de entrada, prediciendo una de las clases si el valor es mayor o igual que 0, y la otra si es menor que 0.

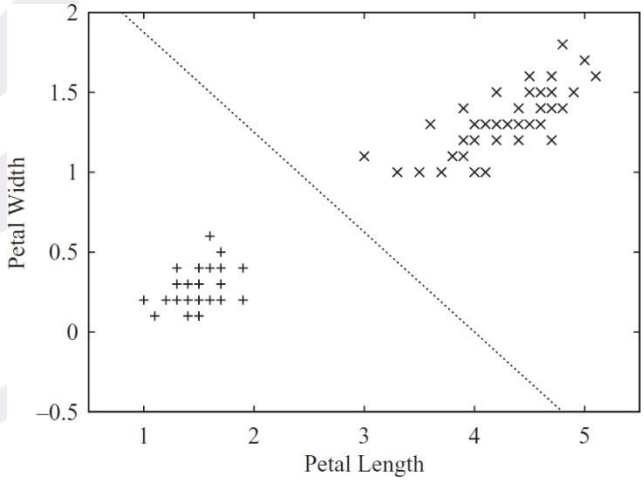

Figura 1. Frontera de decisión en problema de clasificación binaria. Fuente: [Wit11].

En el ejemplo se utilizan dos atributos, por lo que la frontera de decisión es una línea. En caso de utilizar un espacio de 3 dimensiones, la frontera sería un plano, o un hiperplano en casos con mayor dimensionalidad. Por otro lado, en el ejemplo ambas clases son linealmente separables, ya que los datos se pueden separar perfectamente mediante una línea recta, sin cometer errores; sin embargo, en otros casos las clases podrían no ser linealmente separables, y el clasificador trataría de reducir lo máximo posible el error cometido al clasificar una nueva instancia.

Hasta ahora, se ha definido el clasificador lineal para el escenario de clasificación binaria. Para adaptarlo (tanto este como otros clasificadores que se irán viendo en el futuro) al escenario multi-clase, existen dos enfoques principales:

- Uno contra el resto (*one-vs-rest*). En este caso, se entrena un clasificador binario por cada clase, considerando los patrones de una clase como la clase positiva, y el resto de patrones del resto de clases, como la clase negativa.

- Uno contra uno (*one-vs-one*). Se entrena un clasificador por cada par de clases existente, considerando una de ellas como la clase positiva y la otra como la clase negativa. El resto de los patrones no pertenecientes a alguna de las dos clases, se obvian para entrenar dicho clasificador.

En la Figura 2 se observa un ejemplo de clasificador lineal multi-clase *one-vs-rest*. Por ejemplo, la frontera de decisión $y_{1(x)} = 0$ (color azul) separa los puntos de la clase azul del resto de puntos. Por el contrario, en el ejemplo de clasificador *one-vs-one* (Figura 3), la clase azul posee dos fronteras de decisión distintas: $y_{1,2(x)} = 0$ e $y_{1,3(x)} = 0$; la primera separando la clase azul de la naranja, y la segunda separando la clase azul y la verde. Cabe destacar que en este caso con 3 clases, se utilizan el mismo número de clasificadores (o fronteras de decisión) en ambos casos; mientras que cuando el número de clases aumenta, el enfoque *one-vs-one* necesitará de un número de clasificadores lineales mucho más alto que el *one-vs-rest*.

En ambos casos, en la figura de la izquierda vemos las fronteras de decisión por separado. Cabe destacar como en este caso, las clases no son linealmente separables, lo cual se observa más claramente al intentar separar las clases verde y naranja. Posteriormente, al recibir un patrón desconocido, todos los clasificadores lineales proporcionan una salida, y finalmente se asigna a aquella clase con mayor verosimilitud, aquella para la que se obtiene un valor mayor de salida. Por tanto, en la parte derecha de las figuras se observa la partición final de los datos que se produce, dadas las fronteras lineales independientes.

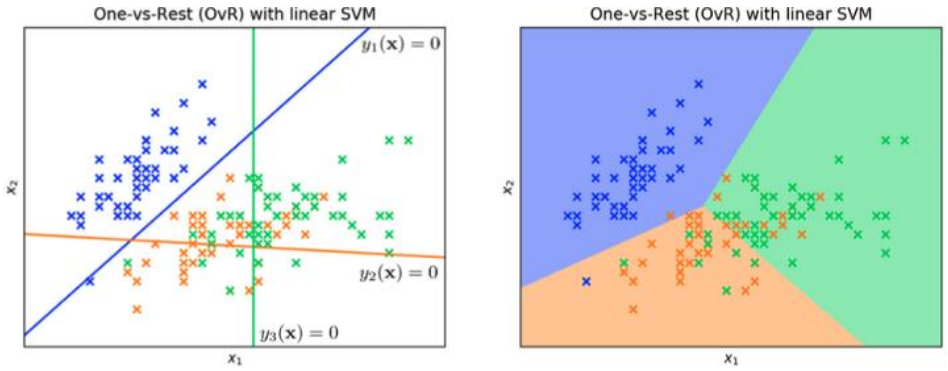

Figura 2. Clasificador lineal one-vs-rest[1].

[1] Fuente: https://ichi.pro/es/ml06-introduccion-a-la-clasificacion-de-clases-multiples-253019598434301

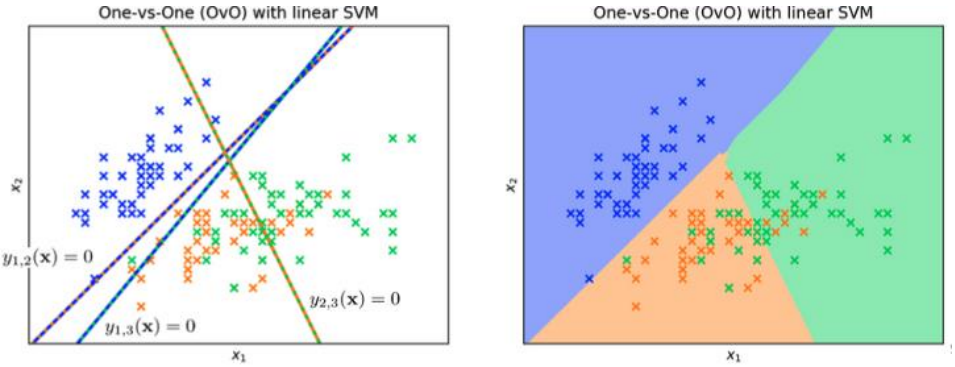

Figura 3. Clasificador lineal one-vs-one[1].

Pese a que la clasificación lineal vista hasta ahora ofrece buenos resultados, también tiene algunos inconvenientes. En primer lugar, el valor de verosimilitud o pertenencia a una clase no son probabilidades, ya que su valor no se encuentra en el rango [0, 1]. Por otro lado, el método de mínimos cuadrados (ampliamente utilizado para calcular el modelo) asume no solo que los errores son estadísticamente independientes, sino que también están distribuidos de manera normal con la misma desviación estándar, asunción que no se cumple en los problemas de clasificación, ya que las observaciones únicamente toman el valor de 0 o 1 en su variable objetivo.

Cabe destacar, por último, que pueden considerarse no solo modelos puramente lineales, sino por ejemplo también cuadráticos. En definitiva, será solo una línea (recta, parábola, etc.) la que separa los datos, a partir de ciertos coeficientes aprendidos. Sin embargo, en esta lección no se entrará en más detalle en estos modelos.

Regresión logística

Para suplir los problemas anteriormente descritos de los modelos clásicos de regresión lineal, surge el modelo de regresión logística, que, pese a su nombre, suele utilizarse para resolver problemas de clasificación. En lugar de aproximar directamente los valores 0 y 1, la regresión logística construye un modelo lineal en base a un valor de salida transformado. En regresión logística se siguen los siguientes pasos:

- Considerando la variable objetivo original $P(1|x_1, x_2, \dots, x_m)$, es decir, la probabilidad de que una instancia descrita por su conjunto de atributos $x_1, \dots, x_n$ sea 1.
- Se convierte en: $\frac{\ln\left(P(1|x_1, x_2, \dots, x_m)\right)}{1-\ln\left(P(1|x_1, x_2, \dots, x_m)\right)}$; de este modo, los valores de la variable de salida ya no están restringidos al intervalo [0, 1], sino que pueden tomar cualquier valor entre $-\infty$ y ∞.

- La variable transformada se aproxima utilizando una función lineal, como las utilizados en la regresión lineal simple. Como resultado, se obtienen los pesos $w_0, w_1, \dots, w_m$.
- El modelo resultante viene dado por la función: $P(1|x_1, \dots, x_m) = \frac{1}{1+\exp(-w_0 - w_1 * x_1 - \cdots - w_m * x_m)}$. La frontera de decisión viene dada, por lo general, por los valores de probabilidad 0.5; es decir, si $P(1|x_1, \dots, x_m) \geq 0.5$, la clase predicha será la clase 1 o clase positiva, y en caso contrario será la clase 0 o clase negativa.

En la Figura 4 se muestra un ejemplo de clasificación con regresión logística. Los datos reales tienen un valor original de 0 o 1, y la función sigmoide (en la que se basa la regresión logística) trata de aproximar dichos valores. Como se observa, los datos no son linealmente separables y por tanto el modelo final cometerá errores sobre los datos de entrenamiento, tratando de minimizar los mismos. El punto en el que $P(Y = 1)$ sea 0.5, es la llamada frontera de decisión, que coincide con el valor de $x = 1.95$. Por tanto, la figura divide los datos reales, mostrando aquellos que se clasifican correctamente (puntos en azul, *accurate classification*), con aquellos en los que se comete error en entrenamiento (cruces en rojo, *innacurate classification*).

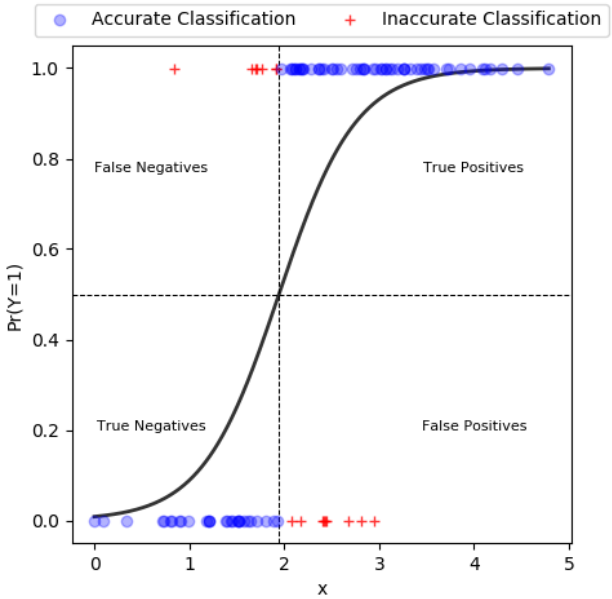

Figura 4. Ejemplo de clasificación con Regresión Logística[2].

[2] Fuente: https://levelup.gitconnected.com/an-introduction-to-logistic-regression-in-python-with-statsmodels-and-scikit-learn-1a1fb5ce1c13

Ejemplos prácticos

En esta sección se trata de presentar algunos problemas, tanto problemas más simples y clásicos para ejemplificar problemas de clasificación, como aplicaciones más complejas de resolución de problemas de clasificación en entornos reales.

5.1. Tiempo meteorológico

El problema del tiempo meteorológico es un conjunto de datos pequeño y ficticio, muy utilizado para ilustrar la resolución de problemas de clasificación. En él, se muestran las supuestas condiciones que son admisibles para jugar un partido de un deporte específico. El *dataset* incluye 4 atributos: *Outlook* (indica si está soleado, lluvioso, o nublado), *Temperature* (temperatura, tomando valores de frío, medio, o cálido), *Humidity* (humedad, con valores de normal o algo), y *Windy* (indica si existe viento); además, el atributo de clase *Play* indica si se puede o no jugar el partido. Como se observa, en este caso todos los atributos son categóricos. En la Tabla 2 se observa el *dataset* al completo, el cual se compone de únicamente 14 instancias.

Tabla 2. Dataset de tiempo meteorológico.

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Este conjunto de datos es simple, pero como se puede observar, todos los datos son de tipo categórico o nominal. Por tanto, sin adaptarlos, no podría aplicarse los métodos de clasificación lineal vistos hasta ahora, que necesitan datos numéricos, pero sí otros métodos que se irán viendo a lo largo del curso. Por ejemplo, un clasificador basado en reglas podría dar un conjunto de reglas[3]

[3] Para más información acerca de los clasificadores basados en reglas, consultar el temario correspondiente más adelante en el curso.

similar al que se muestra a continuación, donde se iría comprobando de manera secuencial hasta que se cumplan las condiciones de una de ellas:

- SI *outlook = sunny* Y *humidity = high* → *play = no*
- SI *outlook = rainy* Y *windy = true* → *play = no*
- SI *outlook = overcast* → *play = yes*
- SI *humidity = normal* → *play = yes*
- SINO → *play = yes*

5.2. Iris

El conjunto de datos Iris[4] es posiblemente el *dataset* más famoso usado en problemas de clasificación. Este conjunto de datos contiene 50 instancias de cada uno de los 3 tipos de planta Iris: Setosa, Versicolor, y Virginica. Tiene 4 atributos, todos ellos numéricos: *sepal length* (largo del sépalo, en cm), *sepal width* (ancho del sépalo, en cm), *petal length* (largo del pétalo, en cm), y *petal width* (ancho del pétalo, en cm); por último, la variable de clase es el tipo de planta Iris que se describe. En la Tabla 3 se muestra un pequeño extracto del conjunto de datos.

Tabla 3. Extracto del dataset Iris.

Sepal length	Sepal width	Petal length	Petal width	Type
5.1	3.5	1.4	0.2	Iris setosa
4.9	3.0	1.4	0.2	Iris setosa
4.7	3.2	1.3	0.2	Iris setosa
		…		
7.0	3.2	4.7	1.4	Iris versicolor
6.4	3.2	4.5	1.5	Iris versicolor
6.9	3.1	4.9	1.5	Iris versicolor
		…		
6.3	3.3	6.0	2.5	Iris virginica
5.8	2.7	5.1	1.9	Iris virginica
7.1	3.0	5.9	2.1	Iris virginica
		…		

En este caso, entre otros clasificadores que se verán a lo largo del curso, también se pueden utilizar los clasificadores lineales vistos, ya que los atributos de entrada son todos numéricos.

Por otro lado, en la Figura 5 se muestra un gráfico donde se dibuja, para cada par de atributos, todas las instancias del dataset en el plano descrito por ambas variables. Como se puede observar, diferenciar la clase Iris Setosa (rojo) del resto, es un problema linealmente separable en

[4] Disponible en: https://archive.ics.uci.edu/ml/datasets/iris

prácticamente todos los casos al escoger 2 de los 4 atributos. Sin embargo, las clases Iris Versicolor e Iris Virginica están más entremezcladas y no son linealmente separables utilizando dos variables (lo que no quiere decir que no lo sean utilizando un espacio de 3 o 4 dimensiones, por medio de un plano o hiperplano, respectivamente).

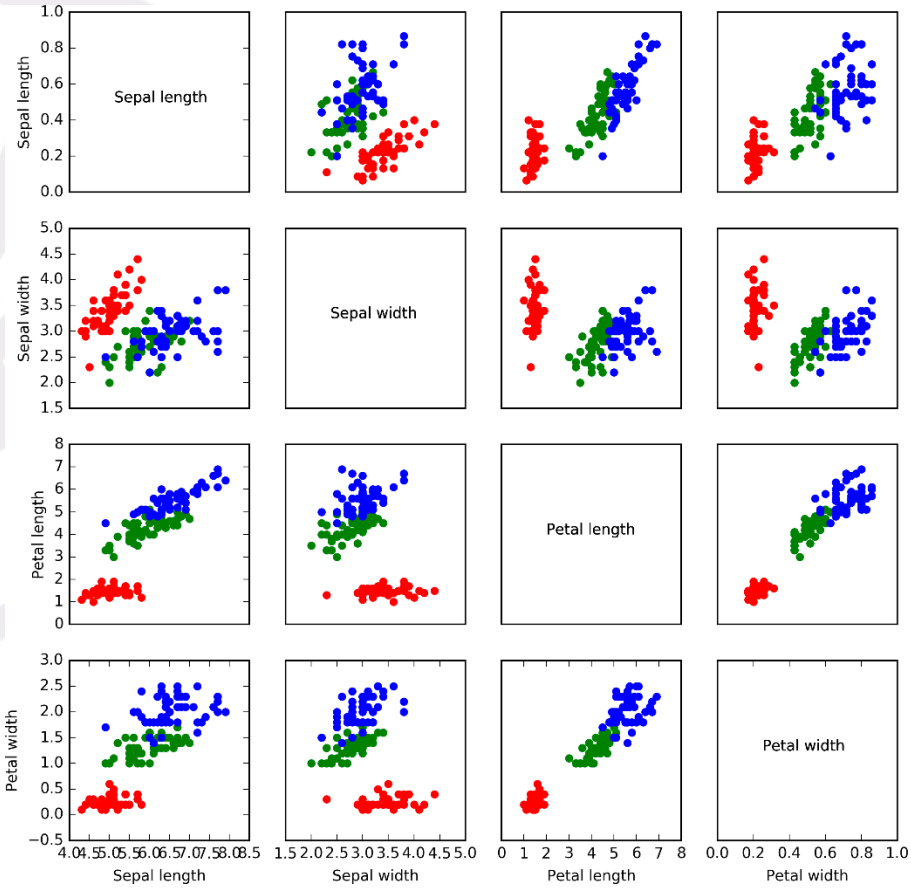

Figura 5. Instancias del dataset Iris dibujadas por cada par de atributos[5].

5.3. MNIST

Pese a que, por lo general, los datos para los algoritmos de clasificación se les muestre en formato tabular, los datos originales pueden provenir de distintas fuentes. El dataset MNIST (*Modified National Institute of Standards and Technology*) [LeC98, LeC10] es uno de los conjuntos de datos más básicos en el ámbito de visión artificial.

[5] Fuente: https://academic.bancey.com/plotting-multivariate-data-with-matplotlibpylab-edgar-andersons-iris-flower-data-set/

El *dataset* MNIST se centra en el reconocimiento de dígitos manuscritos, y está compuesto por un total de 60.000 instancias de entrenamiento y 10.000 de test. Todas las imágenes son todas de 28x28 píxeles, donde el valor numérico está centrado. Mientras que los dígitos de entrenamiento fueron manuscritos por aproximadamente 250 personas, los dígitos de test fueron realizados por personas distintas (pero con perfiles similares a los primeros); así, asegura que los modelos obtenidos son capaces de distinguir dígitos de personas no involucradas en la generación de los datos de entrenamiento.

Se muestra en la Figura 6 un extracto de los dígitos que se contienen en este dataset. Pese a tratarse de imágenes, la gran mayoría de algoritmos de clasificación pueden tratar con este problema. Cada píxel de la imagen es un valor entre 0 y 255 (indicando valores en escala de grises), por lo que el conjunto de datos consta de un total de 28x28=784 atributos numéricos.

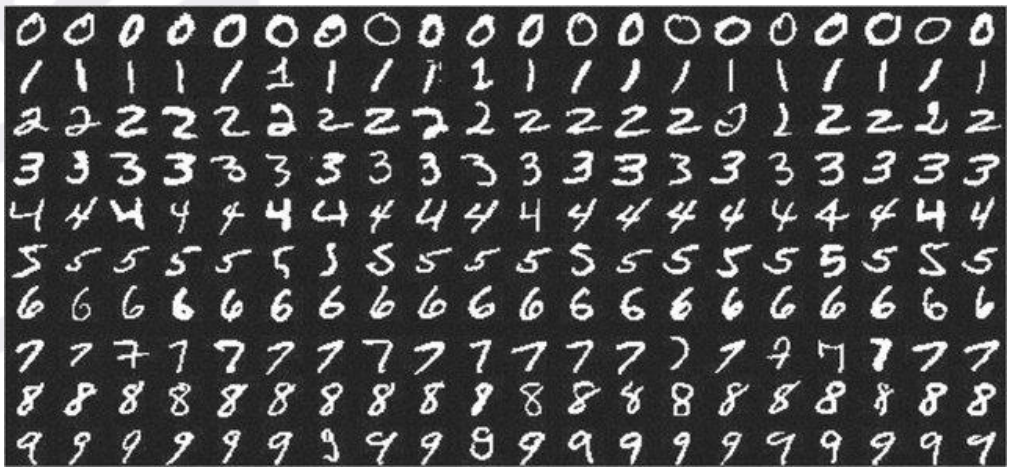

Figura 6. Extracto de los dígitos manuscritos del dataset MNIST[6].

5.4. Aplicaciones reales

Los problemas presentados hasta ahora son ampliamente utilizados para comprender e ilustrar el proceso de resolución de un problema de clasificación, pero son problemas relativamente simples. Sin embargo, las técnicas de clasificación se han utilizado con éxito en multitud de problemas reales a lo largo de los años. Las aplicaciones reales no se limitan, ni mucho menos, a las indicadas a continuación, pero sí que son algunas de las más conocidas.

[6] Fuente: Appiah, K., *et al*. (2009). A binary self-organizing map and its FPGA implementation. DOI: 10.1109/IJCNN.2009.5179001.

Un ejemplo común es en escenarios de marketing, donde la publicidad va enfocada al tipo de cliente objetivo. En estos casos, las variables que describen al cliente se pueden utilizar para predecir sus intereses (variable objetivo) en base a datos previos de entrenamiento.

Recientemente, el uso de técnicas de minería de datos, incluyendo resolución de problemas de clasificación, ha tenido un interés creciente en el campo de diagnosis de enfermedades médicas. En este caso, los atributos de entrada se extraen de la información médica del paciente, y la etiqueta de clase corresponde a si el paciente podrá o no tener una enfermedad concreta en el futuro. Estos problemas deben tratarse con cuidado en muchos casos, ya que puede ser preferible que los errores de clasificación se produzcan al predecir que un paciente tendrá una enfermedad, y que realmente no la tenga en el futuro (falso positivo), que predecir que un cliente no contraerá la enfermedad, y posteriormente la sufra (falso positivo).

En algunos escenarios temporales (donde se tiene en cuenta el paso del tiempo), la etiqueta de clase puede estar asociada con los instantes de tiempo en los que se produce un evento inusual. En estos casos, por lo general son necesarias técnicas específicas para la resolución de problemas basados en series temporales.

El análisis de datos multimedia es en muchas ocasiones complejo porque puede suponer clasificar una gran cantidad de datos multimedia, como imágenes, vídeos, o audio. La complejidad del espacio de características de entrada, y el salto semántico entre los valores de entrada y su significado real hace que dichos problemas se vuelvan un desafío en ocasiones. Otras aplicaciones, como los servicios de noticias online, requieren la clasificación o el filtrado de un gran número de documentos en tiempo real.

En el análisis de datos biológicos, los datos suelen representarse como secuencias discretas, donde se pretende predecir las propiedades de dichas secuencias; o en algunos casos dicha información se proporciona también en forma de grafos o redes. En este escenario, las técnicas de clasificación pueden aplicarse en multitud de diferentes problemas y de distinta manera.

En los últimos años, el análisis de redes sociales se ha vuelto un tema candente. En estos casos, la información está organizada en forma de red, permitiendo, por ejemplo, predecir las etiquetas de nodos relacionados a uno dado. Así, permite predecir las propiedades de actores principales en una red social.

Referencias

[Agg15] Aggarwal, C. C. (2015). Data Classification. Algorithms and Applications. *Chapman and Hall/CRC*.

[Lar14] Larose, D. T., & Larose, C. D. (2014). *Discovering knowledge in data: an introduction to data mining* (Vol. 4). John Wiley & Sons.

[LeC98] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

[LeC10] LeCun, Y., & Cortes, C. (2010). MNIST handwritten digit database. Disponible en: http://yann.lecun.com/exdb/mnist/. Última visita: 23/08/2021.

[Mai10] Maimon, O., & Rokach, L. (Eds.). (2010). Data mining and knowledge discovery handbook, 2nd edition. *Springer*.

[Wit11] Witten, I. H., Frank, E., & Hall, M. A. (2011). Data mining: practical machine learning tools and techniques, 3rd edition. *Morgan Kaufmann*.