ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

GRADO EN INGENIERÍA TELEMÁTICA

PREDICCIÓN DEL ABANDONO DE CLIENTES EN EMPRESAS DE GRAN CONSUMO

Autor: Andrés Díez Montero

Director: Fernando Pavón Pérez

Madrid

Julio 2018

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

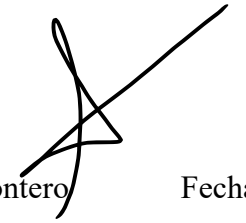
"Predicción del abandono de clientes en empresas de gran consumo"

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2017/18 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: Andrés Díez Montero Fecha: 10/Julio/2018

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Fernando Pavón Pérez Fecha: 10/Julio/2018

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Andrés Díez Montero

DECLARA ser el titular de los derechos de propiedad intelectual de la obra: "PREDICCIÓN DEL ABANDONO DE CLIENTES EN EMPRESAS DE GRAN CONSUMO" que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

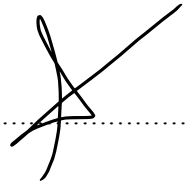
6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 10 de JULIO de 2018

ACEPTA



Fdo……

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

PREDICCIÓN DEL ABANDONO DE CLIENTES EN EMPRESAS DE GRAN CONSUMO

Autor: Díez Montero, Andrés

Director: Pavón Pérez, Fernando

Entidad Colaboradora: GAMCO, S.L.

RESUMEN DEL PROYECTO

En este proyecto se describe el desarrollo de un modelo predictivo basado en técnicas de aprendizaje automático y fundamentalmente, en modelos de Redes Neuronales Artificiales para la predicción del abandono de clientes en empresas de gran consumo. Se ha trabajado con datos reales de las ventas de una empresa de alcance nacional y se han probado diferentes modelos y algoritmos para ver cuál ofrecía un mejor resultado.

Palabras clave: *Aprendizaje automático, LVQ, MCE, SOM, k-NN*

1. INTRODUCCIÓN

Cada cierto tiempo surgen nuevas tendencias en el mundo de la tecnología. Muchas de ellas son modas pasajeras y no llegan a calar en el mercado empresarial. Otras, como la inteligencia artificial, llegan rodeadas de mucha especulación, y aunque su avance pueda no ser explosivo, poco a poco van ganando adeptos.

Muchas empresas han apostado por la Inteligencia Artificial en los últimos años. Según la consultora Gartner [1], el crecimiento del valor de los negocios del sector de la Inteligencia Artificial en 2018 será del 70%. Poniendo más cifras al efecto de esta nueva tendencia, PricewaterhouseCoopers [2] estima que el Producto Interior Bruto (PIB) mundial se incrementará en torno a un 14%, unos 15.7 billones de dólares, en 2030 como consecuencia de los efectos sobre la productividad de la Inteligencia Artificial.

Una de las grandes posibilidades que nos brinda la Inteligencia Artificial es la capacidad de análisis de las grandes cantidades de datos (popularmente conocidos como Big Data) que las empresas llevan almacenando durante años. Esos datos, combinados con técnicas

de Inteligencia Artificial, permitirán hacer un exhaustivo análisis del mercado y obtener grandes ventajas sobre otras empresas de la competencia en prácticamente cualquier sector de mercado. Esto es posible gracias a, entre otras cosas, la capacidad de análisis de información carente de correlación lineal.

2. DEFINICIÓN DEL PROYECTO

El objetivo de este proyecto es el desarrollo de un sistema de predicción basado en técnicas de aprendizaje automático para poder anticiparse al comportamiento de los clientes de empresas de gran consumo; poder predecir un futuro abandono de un cliente para intentar evitarlo y frenar la pérdida que conlleva. Esta predicción deberá realizarse con un mes de antelación para que la empresa tenga el suficiente tiempo de reacción.

Se probarán diferentes algoritmos, ajustando los parámetros de los mismos, para hallar un modelo con la mayor sensibilidad y eficacia posible. Los modelos serán evaluados con diferentes conjuntos de datos para comprobar la robustez de los mismos.

Es importante encontrar el equilibro entre la sensibilidad y la eficacia en los resultados del modelo. Dependiendo del sector del mercado en el que estemos trabajando, una tasa de aciertos lo más cercana a la perfección, a cambio de una menor sensibilidad puede provocar mayor beneficio que una alta sensibilidad.

El proyecto se ha llevado a cabo con datos de una empresa real de nivel nacional. Esta empresa cuenta con gran experiencia en el sector del consumo y es fabricante y distribuidor. Cuenta con aproximadamente 3.600 productos en venta y el proyecto se va a realizar con datos de más de 108.000 clientes. Con esto se quiere destacar que el proyecto se está realizando tiene una aplicación real sobre el mercado y podría tener un importante impacto.

La empresa en cuestión es del sector de la alimentación y distribuye tanto a través del canal HORECA (hoteles, restaurantes y cafeterías) como por los canales de alimentación particular (pequeños supermercados y tiendas) y de grandes supermercados.

3. DESCRIPCIÓN DEL MODELO

A continuación se incluye el diagrama de bloques del proceso seguido para la realización de las pruebas con los diferentes modelos:

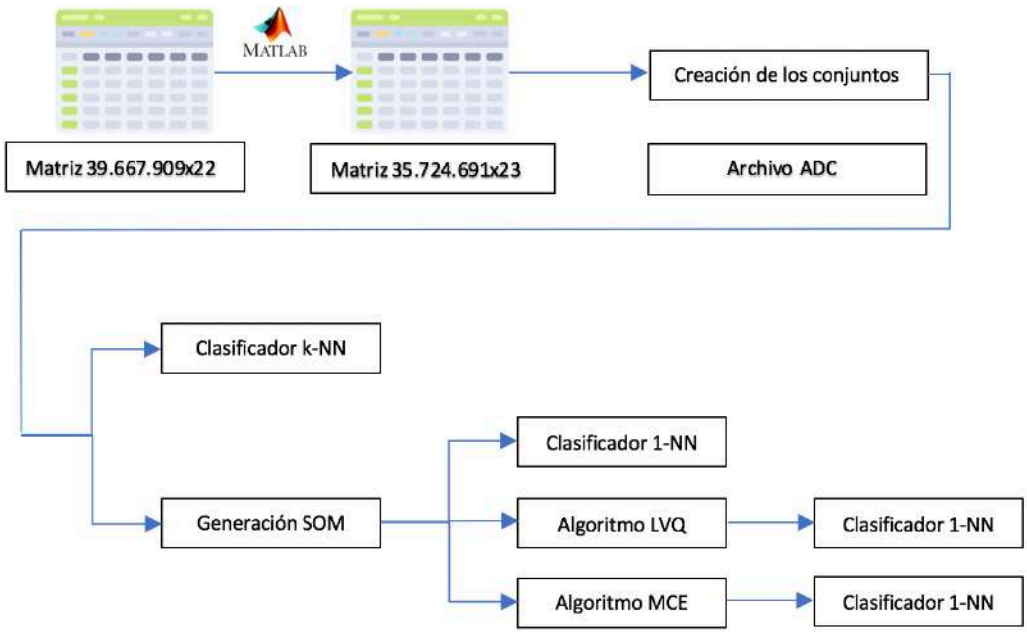


Figura 1 - Diagrama de bloques del proceso de pruebas

Como puede observarse en el diagrama de bloques, el proceso se inicia con una matriz de datos de ventas de las dimensiones indicadas. Esta matriz es analizada para eliminar toda la información que no era relevante para el análisis.

Todo el proceso se ha realizado con la herramienta de software matemático MATLAB.

Una vez se tiene una matriz con información relevante, se crean los conjuntos con los que se entrenan y verifican los modelos. Cuando tengamos los conjuntos definidos se pasa a realizar los experimentos con ellos.

Como se puede observar en el diagrama, primero se usa un clasificador simple sobre los conjuntos para observar los primeros resultados sin la aplicación de algoritmos. Por otro lado, se generan mapas auto-organizados (SOM) que ayudan a la clasificación de los clientes en clases.

El resultado de estos mapas es de nuevo pasado por un clasificador 1-NN para ver los cambios que introduce y posteriormente sometido a los algoritmos LVQ y MCE, cuyas salidas volverán a ser sometidas a un clasificador 1-NN para ser comparados.

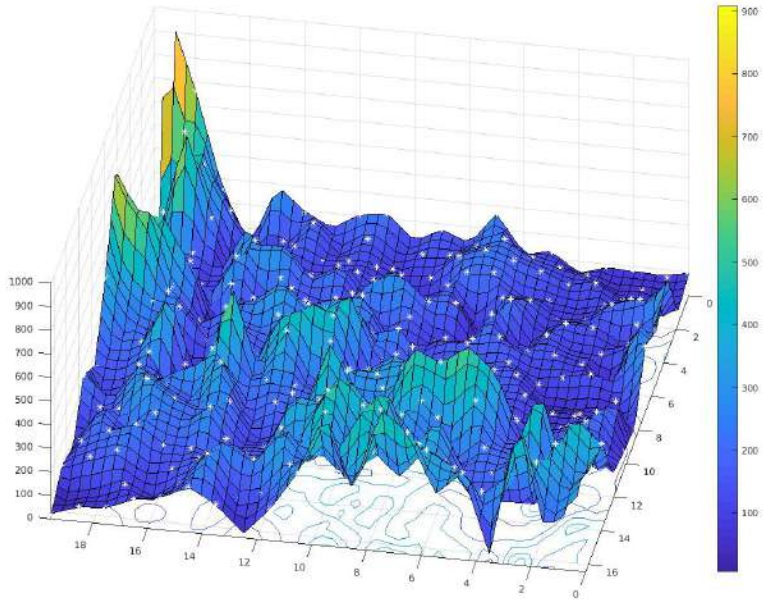


Figura 2 - Mapa auto-organizado de la distribución de clientes

4. RESULTADOS

El objetivo del proyecto era encontrar un modelo que se ajustara bien al problema que se planteaba. También se buscaba medir las diferencias entre los modelos que se iban probando así como la mejora que se obtenía según aumentaba la complejidad de los mismos.

Siguiendo los objetivos del proyecto se han definido las diferentes etiquetas para los resultados:

- **Positivo Cierto:** aquel cliente que se detecta como potencial abandono y resulta acabar siéndolo. Se abreviará como “PC”.
- **Positivo Falso:** aquel cliente que se detecta como potencial abandono y no lo era. Se abreviará como “PF”.
- **Negativo Cierto**: aquel cliente que se considera cliente satisfecho y finalmente es un cliente que no va abandonar a su proveedor. Se abreviará como “NC”.

- **Negativo Falso:** aquel cliente que el modelo piensa que está satisfecho y acaba siendo un abandono para la empresa. Se abreviará como "NF".

Para medir la calidad de los resultados se calculan tres parámetros que se utilizan a lo largo de todo el proyecto:

- **Sensibilidad**: es el número de PC detectados entre la suma de los PC y los NF. Es decir, el porcentaje de clientes que se van que es capaz de detectar.
- **Susceptibilidad:** es el cociente entre los PF y la suma de los PF más los NC. Es decir, el porcentaje de clientes satisfechos que creo el sistema detecta como potenciales bajas.
- **Eficiencia:** el cociente entre PC más NC entre la suma de todos los clientes. Es decir, el porcentaje de aciertos del modelo entre todos los clientes, sumando aciertos en predicción de abandonos y detección de clientes satisfechos.

Como se podía esperar, los últimos resultados fueron mejores que los primeros, obteniéndose los mejores datos para el modelo que utilizaba el algoritmo MCE y clasificado con un clasificador 1-NN.

Para el primer caso de análisis usamos un clasificador k-NN con el que se podían esperar unos resultados positivos pero básicos. De la batería de pruebas con los diferentes parámetros para los conjuntos de prueba, los mejores resultados fueron los siguientes:

Nº prueba	k-NN	Distancia	Sensibilidad	Susceptibilidad	Eficacia
18	5	Ponderado	0.76	0.28	0.74
19	1	Normal	0.72	0.32	0.70
23	5	Normal	0.77	0.29	0.74

Tabla 1 - Parámetros para el clasificador k-NN

La sensibilidad hace referencia al porcentaje de clientes que son potenciales abandonos que se detectan. En el mejor de los casos para la prueba 23, se detectan el 77% de los clientes que pueden dejar de ser clientes. La susceptibilidad hace referencia al porcentaje de falsos positivos dentro de los clientes señalados como posibles abandonos.

Después de generar los mapas auto-organizados, se esperaba una mejoría en los resultados debido a la clasificación previa que se genera en las diferentes neuronas en las que se divide el mapa.

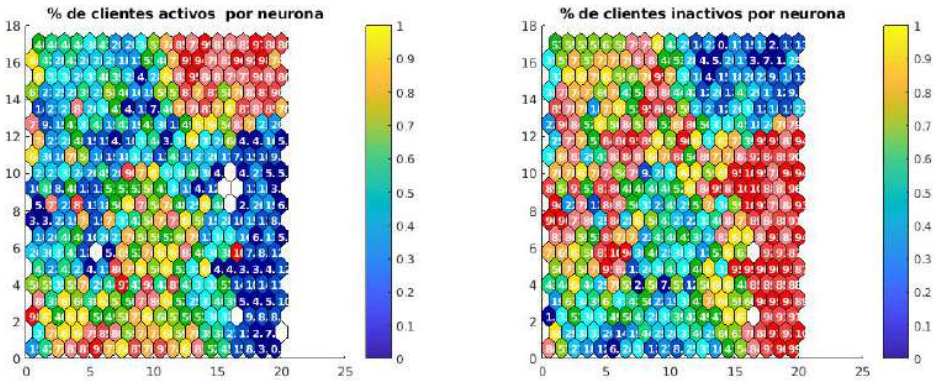


Figura 3 - Mapa auto-organizado de la distribución de los clientes por neurona

Con las SOM generadas, los resultados para el clasificador 1-NN fueron los siguientes:

Nº SOM	Error cuantif.	Sensibilidad	Susceptibilidad	Eficacia
125	4.04	0.84	0.41	0.71
127	4.01	0.83	0.39	0.72
128	3.95	0.86	0.45	0.70

Tabla 2 - Resultados para el clasificador 1-NN después de la SOM

Aunque los resultados son peores que en el primer caso, este resultado no es más que un paso intermedio antes de aplicar los algoritmos que se espera que tengan mejor resultado:

Nº LVQ	Sensibilidad	Susceptibilidad	Eficacia
19	0.81	0.28	0.76
22	0.80	0.27	0.76
28	0.82	0.28	0.77

Tabla 3 - Resultados algoritmo LVQ+1-NN

Nº MCE	Sensibilidad	Susceptibilidad	Eficacia
66	0.80	0.23	0.77
71	0.81	0.23	0.76
72	0.81	0.24	0.76

Tabla 4 - Resultados algoritmo MCE+1-NN

Se puede observar que ambos resultados fueron similares en torno a sensibilidad y eficacia pero el modelo MCE mejora a LVQ en el porcentaje de falsos positivos. Ambos modelos mejoran claramente el primer clasificador k-NN.

5. CONCLUSIONES

El objetivo era generar modelos que proporcionaran los mejores resultados posibles para la predicción del abandono de clientes. Los resultados no son tan buenos como inicialmente se había pensado que podrían ser, pero se ha conseguido mejoría con los últimos experimentos realizados.

La sensibilidad ha sido un parámetro que en algunos casos de las baterías de pruebas tenía valores realmente altos, sin embargo, es importante elegir aquellos experimentos que también tengan una baja tasa de falsos positivos, ya que de esta forma aumenta la calidad del modelo en conjunto.

6. REFERENCIAS

[1] Dealer World, "Gartner estima que el Mercado global de la IA sobrepasará el billón de dólares", abril 2018. http://www.dealerworld.es/mercado-en-cifras/gartner-estima-que-el-mercado-global-de-la-ia-sobrepasara-el-billon-de-dolares

[2] Severino, Carlos. Ideas PwC, "Si, le ponemos cifras a la Inteligencia Artificial", julio 2017. http://ideas.pwc.es/archivos/20170714/le-ponemos-cifras-a-la-inteligencia-artificial

7. ÍNDICE DE FIGURAS

8. ÍNDICE DE TABLAS

CLIENT ABANDONMENT PREDICTION IN LARGE CONSUMER-SECTOR COMPANIES

Author: Díez Montero, Andrés

Supervisor: Pavón Pérez, Fernando

Collaborating Entity: GAMCO, S.L.

ABSTRACT

This project describes the development of a predictive model based on machine learning techniques and specially in Artificial Neural Networks. The project has been carried out with real data from a national-range sales company. Different models and algorithms have been tested to see which one offered a better result.

Keywords: *Machine learning, LVQ, MCE, SOM, k-NN*

1. INTRODUCTION

In technology world, we have new trends from time to time. Most of them end up being a passing fad. Others, like Artificial Intelligence appear surrounded by great speculation, and although their development might not be very fast, the end up having a lot of supporters.

Many companies have invested in Artificial Intelligence during the last few years. According to consultant Gartner [1], the increase in the value of Artificial Intelligence sector businesses in 2018 will be 70%. PricewaterhouseCoopers [2] has estimated that the global GDP will be 14% higher (15.7 billion dollars) by 2030 thanks to the effects of Artificial Intelligence on productivity.

One of the big possibilities Artificial Intelligence bring us, is the ability to analyze the huge amounts of data (popularly known as Big Data) that companies have been storing during years. This data combined with Artificial Intelligence techniques will allow them to analyze their market and gain advantages over competitors. This is possible thanks to the ability to analyze non-correlated information.

2. PROJECT DEFINITION

The goal of this project is the development of prediction model based on machine learning techniques, to try to predict the behavior of consumer-sector companies clients. Being able to predict when a client is going to stop buying gives us time to try to change his mind.

The project will analyze different algorithms changing their parameters to find the model with the highest sensitivity and effectiveness. Models will be tested with different data samples to test their sturdiness.

It is important to find balance between sensitivity and effectiveness in the model. Depending on the market sector we are working at, a close-to-perfect effectiveness rate in exchange for a lower sensitivity, might have higher profit than a higher sensitivity.

The project has been done with data from a first-level national company. This company has great experience in the consumer sector as manufacturer and distributor of their products. It sells more than 3.600 products and the data for the project has more than 108.000 clients. This information is to highlight the fact that the project has been made with real data form a real company and could have a big impact on the sector.

This company sells products to hotels, restaurants, bars, small and big supermarkets.

3. MODEL DESCRIPTION

Here is a diagram of the process followed to create and test the models:

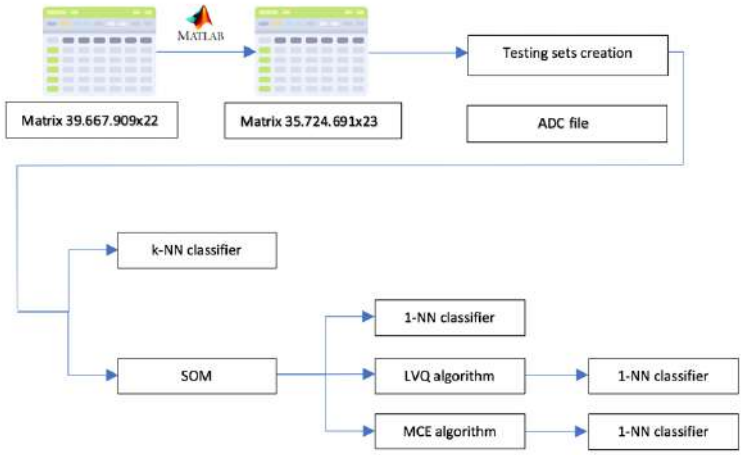


Illustration 1 - Block diagram of the model-testing process

The block diagram shows that the process starts with a raw sales rata matrix. The matrix is analyzed to remove all the non-relevant information.

The whole process has been carried out with the mathematics software tool MATLAB.

Once we have the processed-data matrix, we create the data sets that are used to train and verify the models. Once the data sets have been created, we can start with the tests.

As you can see in the block diagram, the first step is a classifier. This classifier allows us to see the first results without using any algorithms. After this, we generate self-organized maps (SOM) that help identify the different client classes.

A 1-NN classifier is used to test the SOM classification quality. We run LVQ and MCE algorithms on the SOMs and after that, we check the improvements in the results with the 1-NN classifier that helps us compare the output parameters.

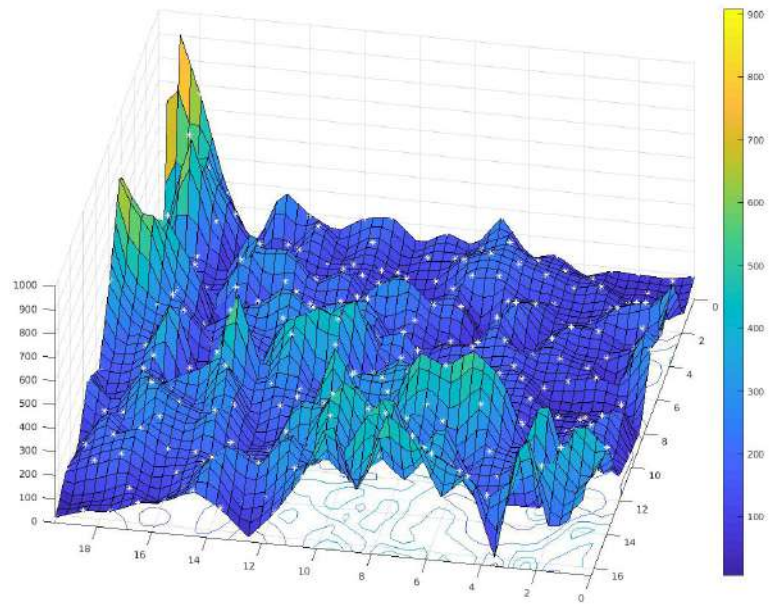


Illustration 2 - Clients distribution self-organized map

4. RESULTS

The challenge of the project was to find a model that fitted the specific business situation we were working with. We wanted to check the differences between the models and measure how the results improved as the models became more complex.

Following the goals of the project, the following tags have been defined:

- **True Positive:** the system thinks a client can leave my Company and he finally does. From now on we will talk about “TP”.
- **False Positive:** the system thinks a client can leave my company but he is happy with the received attention. From now on we will talk about “FP”.
- **True Negative**: the system thinks a client wants to stay with my company and he finally does. From now on we will talk about “TN”.
- **False Negative:** the model thinks a client wants to stay with my company but he wants to stop buying my products. From now on we will talk about “FN”.

Three parameters have been defined to measure the quality of the models:

- **Sensitivity**: TP/(TP+FN). Percentage of clients that leave the system can detect.
- **Susceptibility:** FP/(FP+TN). Percentage of happy clients that the system thinks want to leave me**.**
- **Effectiveness:** (TP+TN)/(TP+TN+FP+FN). Percentage of clients the system can detect properly (both positive and negative).

The results from the last models were better that the first ones as we were expecting. The best results were obtained for the MCE algorithm with the 1-NN classifier.

For the first test we used a k-NN classifier. Although we were not expecting great results, they were quite positive. The best output parameters after several tests with the input parameters of the k-NN classifier were:

Test nº	k-NN	Distance	Sensitivity	Susceptibility	Effectiveness
18	5	Weighted	0.76	0.28	0.74
19	1	Normal	0.72	0.32	0.70
23	5	Normal	0.77	0.29	0.74

Chart 1 - k-NN classifier output parameters

Sensitivity is a parameter referred to the percentage of clients that might stop buying my products the model can detect. The best sensitivity was for test 23, with the 77% of potential abandonments detection. Susceptibility is an output parameter that refers to the percentage of false-positive abandonment predictions.

After generating the self-organized maps, we were expecting a quality increase in the models due to the neural classification of the SOM.

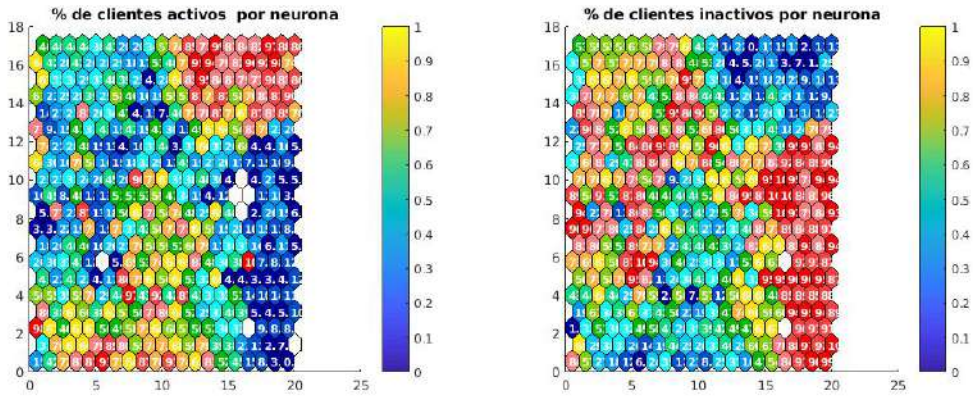


Illustration 3 - Neural clients distribution SOM

After generating the SOM, the results with the 1-NN classifier are shown next:

SOM nº	Cuantif. error	Sensitivity	Susceptibility	Effectiveness
125	4.04	0.84	0.41	0.71
127	4.01	0.83	0.39	0.72
128	3.95	0.86	0.45	0.70

Chart 2 - SOM+1-NN classifier output parameters

The results are clearly worse than the previous ones. Although this is not something positive, the SOM is just and intermediate step before using LVQ and MCE algorithms, so a big improvement is expected at the end of the process.

LVQ nº	Sensitivity	Susceptibility	Effectiveness
19	0.81	0.28	0.76
22	0.80	0.27	0.76
28	0.82	0.28	0.77

Chart 3 - LVQ+1-NN classifier output parameters

MCE nº	Sensitivity	Susceptibility	Effectiveness
66	0.80	0.23	0.77
71	0.81	0.23	0.76
72	0.81	0.24	0.76

Chart 4 - MCE+1-NN classifier output parameters

As we can see, sensitivity and effectiveness are similar for both algorithms. MCE has a lower susceptibility than LVQ (false-positive rate). Both algorithms improve the first k-NN classifier results.

5. CONCLUSIONS

The original goal was to generate models to provide the best possible predictive results. The final output parameters are not as good and efficient as we had thought they could be. Nevertheless, the process has shown improvements as we have tested different algorithms and techniques. The improvement between the first and the last results is remarkable.

Some of the tests had a much higher sensitivity, nevertheless, all the tests that had a sensitivity this high had a lack of effectiveness. That is why we have tried to find the balance between both parameters so that the quality of the model as a whole increases.

6. REFERENCES

[1] Dealer World, "Gartner estima que el Mercado global de la IA sobrepasará el billón de dólares", April 2018. http://www.dealerworld.es/mercado-en-cifras/gartner-estima-que-el-mercado-global-de-la-ia-sobrepasara-el-billon-de-dolares

[2] Severino, Carlos. Ideas PwC, "Si, le ponemos cifras a la Inteligencia Artificial", July 2017. http://ideas.pwc.es/archivos/20170714/le-ponemos-cifras-a-la-inteligencia-artificial

7. ILLUSTRATION INDEX

8. CHART INDEX

ÍNDICE DE LA MEMORIA

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

1.1 MOTIVACIÓN DEL PROYECTO

Este proyecto Fin de Carrera surge de una propuesta de colaboración de GAMCO, S.L.

GAMCO es una compañía dedicada a la creación de soluciones software basadas en modelos predictivos obtenidos a partir de la Inteligencia Artificial y el Aprendizaje Automático. El proyecto discurre en la línea de trabajo de GAMCO: creación de modelos predictivos para conseguir que empresas de primera línea puedan aumentar sus ingresos, mejorar el servicio a los clientes y optimizar recursos.

En el caso de este proyecto Fin de Carrera, se busca la creación de un sistema de predicción del abandono de clientes en las empresas de gran consumo. La fidelización de clientes es un tema crucial para las empresas del sector del consumo. Estamos viviendo unos años de consumo salvaje, en los que la gran cantidad de alternativas y la variedad en los rangos de precios hacen que maximizar los esfuerzos por mantener satisfechos a los clientes sea un sinónimo de éxito empresarial.

Sin embargo, para algunas de estas empresas, debido al elevado número de clientes y a la variedad de productos que ofertan, aun sabiendo que cada cliente tiene unas necesidades diferentes, es muy complicado ofrecer la atención personalizada que haga que todo el mundo quede satisfecho. Es por eso, que muchas veces los clientes acaban rotando entre empresas del mismo sector, buscando mejores ofertas o productos nuevos que se adapten más a sus necesidades en cada momento.

En este proyecto se quiere estudiar el comportamiento que tienen tanto los clientes satisfechos como los clientes que son potenciales abandonos de una marca, intentando establecer patrones de comportamiento para ambos grupos de manera que pueda predecirse cuando un cliente del primer grupo pueda pasarse al segundo.

De esta manera, si nuestro sistema es capaz de predecir cuando el comportamiento de un cliente empieza a mostrar signos de descontento con el servicio o la situación en la que se encuentra, poder, de manera personalizada, llegar hasta él actuar en consecuencia, ya sea a través de ofertas, promociones, descuentos, etc.

Cualquier empresa con una herramienta de este estilo adquiere una ventaja considerable sobre sus competidoras, ya que cuando eres capaz de anticiparte a una situación como que un cliente decida cambiar de proveedor, el beneficio obtenido de mantener a ese cliente es mucho mayor que el coste de conseguirlo.

En el mercado actual es altamente competitivo y ofrece tantas posibilidades, que la diferencia entre el éxito o el fracaso se esconde en unos pequeños detalles. Por eso, muchas empresas están dedicando esfuerzo y recursos en la mejora de sus servicios con ayuda de las herramientas que la Inteligencia Artificial ofrece.

El proyecto se ha llevado a cabo con datos de una empresa real de nivel nacional. Esta empresa cuenta con gran experiencia en el sector del consumo y es fabricante y distribuidor. Cuenta con aproximadamente 3.600 productos en venta y el proyecto se va a realizar con datos de más de 108.000 clientes. Con esto se quiere destacar que el proyecto se está realizando tiene una aplicación real sobre el mercado y podría tener un importante impacto.

La empresa en cuestión es del sector de la alimentación y distribuye tanto a través del canal HORECA (hoteles, restaurantes y cafeterías) como por los canales de alimentación particular (pequeños supermercados y tiendas) y de grandes supermercados.

CAPÍTULO 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 CLASIFICADOR K-NN

K-NN [1] es un método de clasificación básico que se ha utilizado en diferentes partes del proyecto. Su funcionamiento es sencillo y se resume en que cada nuevo caso que se analice va a ser reconocido como parte de la clase a la que pertenezcan sus K elementos vecinos más cercanos.

El clasificador k-NN tiene un aprendizaje basado en casos o instancias, es decir analiza casos conocidos y los utiliza para clasificar elementos nuevos. Esto también se conoce como aprendizaje supervisado. Los datos como los que se alimenta al clasificador en la parte de aprendizaje constan de los atributos de cada elemento y la clase a la que pertenecen. En el caso particular de este proyecto, se trata de un problema biclase (0,1).

Para el correcto entrenamiento y verificación del modelo, los conjuntos de datos se dividieron en conjuntos de entrenamiento, conjuntos de prueba y conjuntos nuevos.

$$D_{m,n} = \begin{bmatrix} d_1 = & a_{1,1} & \cdots & a_{1,n} & c_1 \\ d_2 = & a_{2,1} & \ddots & a_{2,n} & c_2 \\ \vdots & \vdots & \vdots & \vdots & \\ d_m = & a_{m,1} & \cdots & a_{m,n} & c_m \end{bmatrix}$$

Figura 1 - Matriz del conjunto de datos de un clasificador k-NN

Siendo *D* nuestro conjunto de datos, con *m* el número de clientes, *n* el número de atributos de cada cliente y *c* la clase a la que pertenecen.

Una vez se ha producido el entrenamiento, introducimos una nueva instancia del conjunto de pruebas *d,* y en función del parámetro K que hayamos escogido para el algoritmo, la nueva instancia se reconocerá como miembro de la clase que tenga más cercana.

Es un clasificador muy simple por lo que no hay muchos parámetros que podamos variar para afinarlo. Podemos destacar que es muy lento, por lo que hacerlo directamente sobre los conjuntos de datos puede resultar una tarea muy pesada y puede ser preferible utilizarlo después de crear una SOM y aplicar algún otro algoritmo. Los parámetros más importantes son:

- *k:* se refiere al número de vecinos con los que se compara la nueva instancia que se quiere clasificar. No hay ninguna norma acerca de que un mayor número de vecinos consiga mejores resultados, hay que analizarlo para cada situación particular.
- Métrica: es la distancia de similitud con las instancias del modelo. La métrica más utilizada es la distancia Euclídea.
- Votación: este parámetro se refiere al sistema empleado para la votación de los vecinos más cercanos. Una vez tenemos las distancias de los *k* vecinos más cercanos, puede que una instancia se asigne a una clase por haber dos elementos de esa clase contra uno de otra pero que sin embargo, ese elemento sea mucho más cercano. Por tanto, la votación para la elección de la clase puede ser por el número de vecinos, o ponderando el número de vecinos de cada clase por la distancia de cada uno a la nueva instancia.

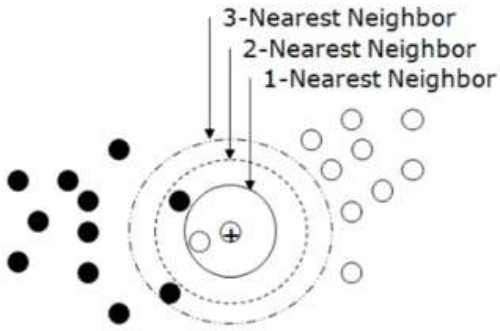


Figura 2 - Imagen que ilustra la selección de una nueva instancia por sus vecinos

2.2 MAPAS AUTO-ORGANIZADOS

Los mapas auto-organizados [2], popularmente conocidos como *SOMs*, son sistemas de aprendizaje no supervisado competitivo. Como tal, la SOM no ofrece un resultado cuantificable sobre la calidad de la clasificación, por eso, en el proyecto, se ha usado el algoritmo 1-NN para verificar la calidad de las SOMs.

En el entrenamiento de la SOM, a diferencia del clasificador k-NN, solo se conocen las entradas. Durante el proceso de entrenamiento, la red tiene que descubrir patrones entre los datos de entrada e incorporarlos al vector de pesos de cada neurona.

La idea de aprendizaje competitivo es la siguiente: cada vez que se presenta una nueva instancia, se calcula la distancia euclidiana a todos los vectores de pesos. La neurona que tenga el vector de pesos más similar se convierte en la BMU (*best matching unit*). Después, los pesos de esta neurona y los de las cercanas a ella son ajustados hacia el vector de entrada. Inicialmente los cambios son muy notables pero con el paso del entrenamiento se vuelven más suaves.

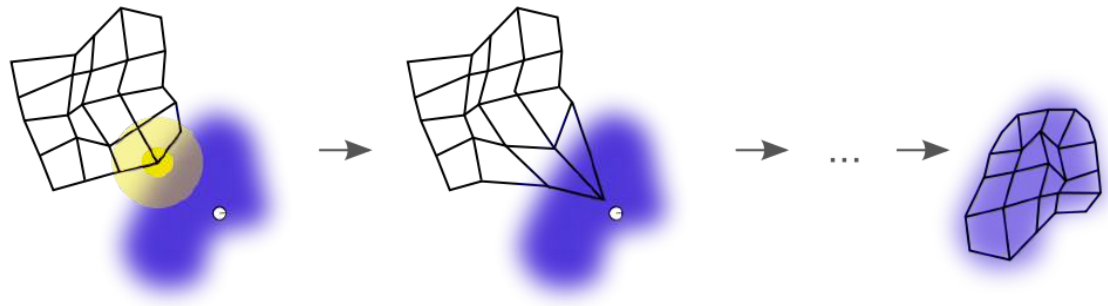


Figura 3 - Evolución del ajuste de los pesos con el paso de las iteraciones

Los pasos en la generación de la SOM son los siguientes:

- Se genera un mapa con vectores de pesos aleatorios.
- Se toma un vector de entrada D(t)
    - Se itera por cada neurona del mapa
        - Se calculan las distancias entre el vector de entrada y los vectores de pesos de las neuronas del mapa.

- La neurona con la menor distancia es la BMU.

- Se actualizan las neuronas vecinas de la BMU.

$$W_v(s+1) = W_v(s) + \Theta(u, v, s)\alpha(s)(D(t) - W_v(s))$$

Figura 4 - Fórmula del ajuste de los pesos de las neuronas vecinas a la BMU

- Se pasa a la siguiente iteración salvo que se haya superado el número de iteraciones establecido.

Los parámetros de la fórmula de la actualización son los siguientes:

- *s:* es la iteración que se está llevando a cabo.
- *λ:* es la cantidad total de iteraciones.
- *t:* índice del vector de entrada.
- *D(t):* vector de entrada.
- *v:* índice de una neurona de la SOM.
- *Wv:* vector de pesos de la neurona v.
- *U:* índice de la neurona BMU.
- *Θ(u,v,s):* función de vecindad.
- *α(s):* restrictor de aprendizaje por el progreso de las iteraciones.

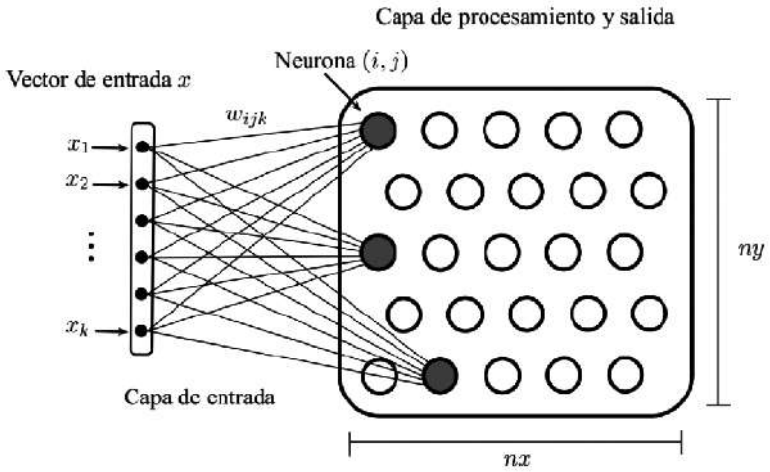


Figura 5 - Esquema del ajuste de pesos de la SOM para un vector de entrada [3]

2.3 ALGORITMO LVQ

El algoritmo LVQ [4] (*Learning Vector Quantization*) es un método de cuantificación de vectores supervisado propuesto por Kohonen para diseñar los prototipos de clasificación 1-NN usando reglas de corrección de errores.

El funcionamiento es el siguiente: todos los patrones de entrenamiento son enviados al clasificador NN repetidamente para actualizar los prototipos. Para un patrón de entrada x, se encuentran los dos los dos prototipos más cercanos (m_i y m_j). Si m_i pertenece a la verdadera clase de x mientras que m_j pertenece a otra clase y x cae en una ventana (la distancia de ambos prototipos a x es comparable)

$$\min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i}\right) > \frac{1-w}{1+w},$$

Figura 6 - Ecuación de comparación de distancia de prototipos

donde w es la anchura relativa de la ventana, los prototipos son actualizados según:

$$\mathbf{m}_i = \mathbf{m}_i + \alpha(t)(\mathbf{x} - \mathbf{m}_i),$$
$$\mathbf{m}_j = \mathbf{m}_j - \alpha(t)(\mathbf{x} - \mathbf{m}_j),$$

Figura 7 - Ecuación de actualización de prototipos con LVQ

Donde α(t) es la tasa de aprendizaje, que es suficientemente pequeña y decrece con el tiempo. Con las actualizaciones, m_i es acercado a x mientras que m_j es alejado.

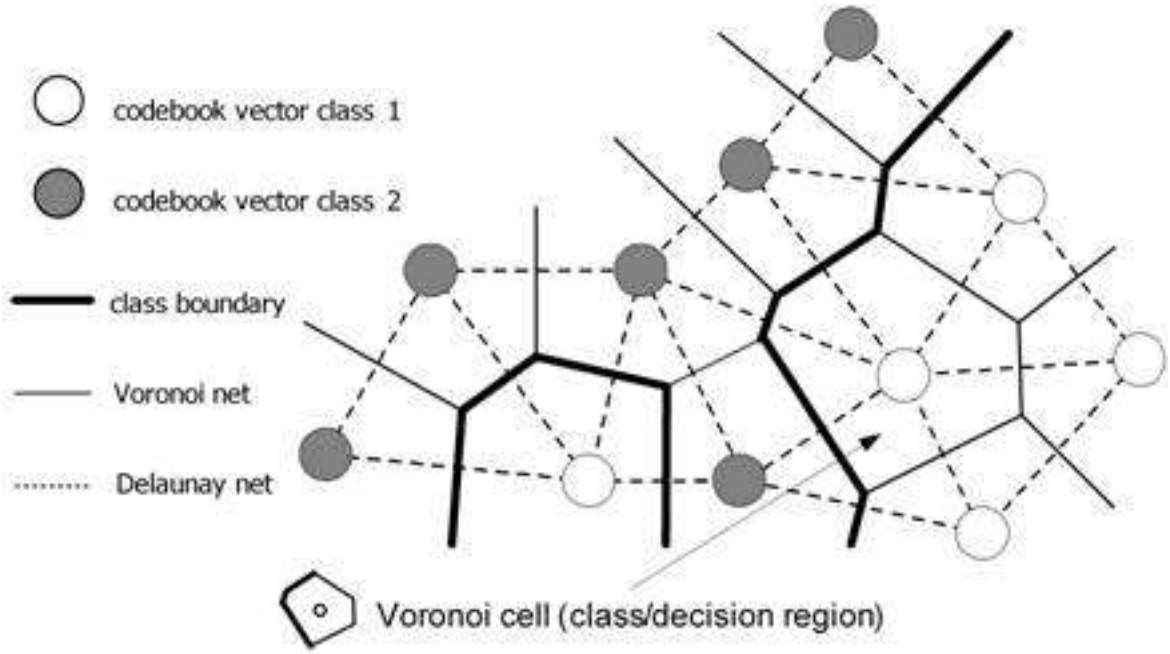


Figura 8 - Distribución de los elementos de dos clases en una SOM [5]

2.4 ALGORITMO MCE

El algoritmo MCE [6] fue propuesto inicialmente por Biing-Hwang Juang y S. Katagari como un método de entrenamiento de clasificación mínima de errores para el aprendizaje de parámetros de redes neuronales y clasificadores estadísticos.

Definieron una función de pérdida basada en funciones discriminantes y minimizaron la pérdida empírica en una muestra de entrenamiento establecida por descenso de gradiente para optimizar

los parámetros de clasificación. En el caso del clasificador 1-NN, la función discriminante de una clase es el negativo de la distancia mínima desde el patrón de entrada a esta clase es:

$$g_k(\mathbf{x}) = -\min_j d(\mathbf{x}, \mathbf{m}_{kj}).$$

Figura 9 - Función discriminante del clasificador 1-NN para MCE

La medida del error de clasificación de un patrón de la clase k viene dado por la función:

$$\mu_k(\mathbf{x}) = -g_k(\mathbf{x}) + \left[\frac{1}{M-1}\sum_{j\neq k} g_j(\mathbf{x})^\eta\right]^{1/\eta},$$

Figura 10 - Función de la medida del error de clasificación de un patrón de una clase

Donde η es un número positivo. Cuando el valor de η tiende a infinito, la medida del error de clasificación se mide por la siguiente ecuación:

$$\mu_k(\mathbf{x}) = -g_k(\mathbf{x}) + g_r(\mathbf{x}),$$

Figura 11 - Función de la medida del error de clasificación cuando η tiende a infinito

En la que $g_r(x)$ es la función discriminante de la clase rival más cercana definida por:

$$g_r(\mathbf{x}) = \max_{i\neq k} g_i(\mathbf{x}).$$

Figura 12 - Función discriminante de la clase rival más cercana

Se ve fácilmente que cuando el patrón de entrada es clasificado erróneamente (la función discriminante de la clase k no es el máximo), la medida $\mu_k(x)$ es positiva, de lo contrario, será negativa. En el caso del calificador 1-NN, la medida del error de clasificación viene dada por:

$$\mu_k(\mathbf{x}) = d(\mathbf{x}, \mathbf{m}_{ki}) - d(\mathbf{x}, \mathbf{m}_{rj}),$$

Figura 13 - Función de la medida del error de clasificación

Donde m_{ki} es el prototipo más cercano a clase correcta mientras que m_{rj} (de la clase r) es el prototipo más cercano de la clase incorrecta. La pérdida asociada con la clasificación del patrón x viene dada por:

$$l_k(\mathbf{x}) = l_k(\mu_k) = \frac{1}{1 + e^{-\xi\mu_k}}.$$

Figura 14 - Función de la pérdida asociada a la clasificación de un patrón

En una muestra de un co0njunto de entrenamiento, la pérdida empírica promedio se calcula mediante:

$$L_0 = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{M} l_k(\mathbf{x}^n) I(\mathbf{x}^n \in C_k),$$

Figura 15 - Función de la pérdida empírica promedio

Donde I(.) es una función indicador que toma el valor de uno cuando la condición de los paréntesis se cumple y un cero si no se cumple.

Para minimizar la pérdida empírica mediante la búsqueda de gradientes estocásticos, los prototipos se actualizan alimentando un patrón de entrenamiento. Como la función de pérdidas $l_k(x)$ involucra dos prototipos (m_{ki} y m_{rj}), solo los dos prototipos se actualizan en el patrón de entrenamiento x.

$$\mathbf{m}_{ki} = \mathbf{m}_{ki} - \alpha(t)\frac{\partial l_k(\mathbf{x})}{\partial \mathbf{m}_{ki}},$$
$$\mathbf{m}_{rj} = \mathbf{m}_{rj} - \alpha(t)\frac{\partial l_k(\mathbf{x})}{\partial \mathbf{m}_{rj}}.$$

Figura 16 - Ecuaciones de la medida del error de clasificación

Calculando las derivadas parciales:

$$\frac{\partial l_k(\mathbf{x})}{\partial \mathbf{m}_{ki}} = 2\xi l_k(1 - l_k)(\mathbf{m}_{ki} - \mathbf{x}),$$
$$\frac{\partial l_k(\mathbf{x})}{\partial \mathbf{m}_{rj}} = 2\xi l_k(1 - l_k)(\mathbf{x} - \mathbf{m}_{rj}).$$

Figura 17 - Ecuaciones derivadas parciales de la medida del error de clasificación

Por último, para obtener la regla de actualización combinamos las ecuaciones anteriores:

$$\mathbf{m}_{ki} = \mathbf{m}_{ki} + 2\alpha(t) l_k(1 - l_k)(\mathbf{x} - \mathbf{m}_{ki}),$$
$$\mathbf{m}_{rj} = \mathbf{m}_{rj} - 2\alpha(t) l_k(1 - l_k)(\mathbf{x} - \mathbf{m}_{rj}),$$

Figura 18 - Regla de actualización MCE

En la que ξ ha sido absorbido por la tasa de aprendizaje $\alpha(t)$.

CAPÍTULO 3. ESTADO DE LA CUESTIÓN

En los últimos años hemos estado viviendo una tendencia de las empresa a almacenar grandes cantidades de datos. Anteriormente, los datos eran registrados de una forma más o menos desestructurada, pero con la aparición de la nube y de los nuevos recursos que la Inteligencia Artificial ("IA") brinda, esta información ha comenzado a ganar valor y a representar un nuevo filón de negocio.

3.1 PRINCIPALES APLICACIONES DE LA INTELIGENCIA ARTIFICIAL

Aunque el término Inteligencia Artificial aún nos suene un poco futurista, lo cierto es que a día de hoy entramos en contacto con algunas de sus aplicaciones casi a diario. Entre las más utilizadas se encuentran [7]:

- Generación de lenguaje natural: es una sub-disciplina de la Inteligencia Artificial que busca convertir los datos en texto, permitiendo a los ordenadores comunicar ideas de forma precisa y realista.
- Reconocimiento de voz: son los sistemas que transcriben el lenguaje humano. Aunque el ejemplo que primero nos viene a la cabeza es *Siri* (propiedad de Apple Inc.), existen en la actualidad multitud de asistentes por voz que mejoran a pasos agigantados día a día.

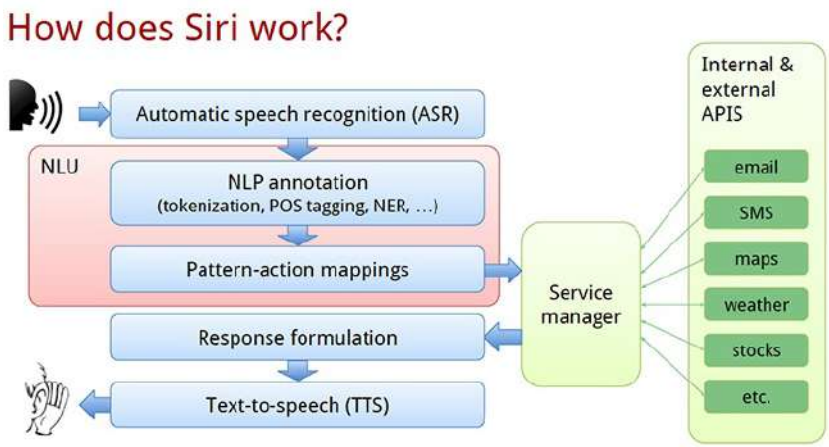


Figura 19 - Esquema de funcionamiento del asistente virtual "Siri" [8]

- Agentes virtuales: sistemas capaces de interactuar con humanos. Su calidad es aún limitada, pero están siendo muy utilizados en sistemas de atención al cliente.
- Toma de decisiones: muy similar a lo que se hace en este proyecto. Se introducen unas reglas a los sistemas de Inteligencia Artificial que a partir de un entrenamiento puede ayudar a la toma de decisiones que ayude a mejorar la rentabilidad.
- Herramientas biométricas: es la tecnología capaz de identificar medir y analizar el cuerpo humano. Uno de los principales usos de estas herramientas biométricas es para la seguridad en los desbloqueos de los teléfonos inteligentes, tanto por rasgos faciales, como por reconocimiento ocular, como por huella dactilar.

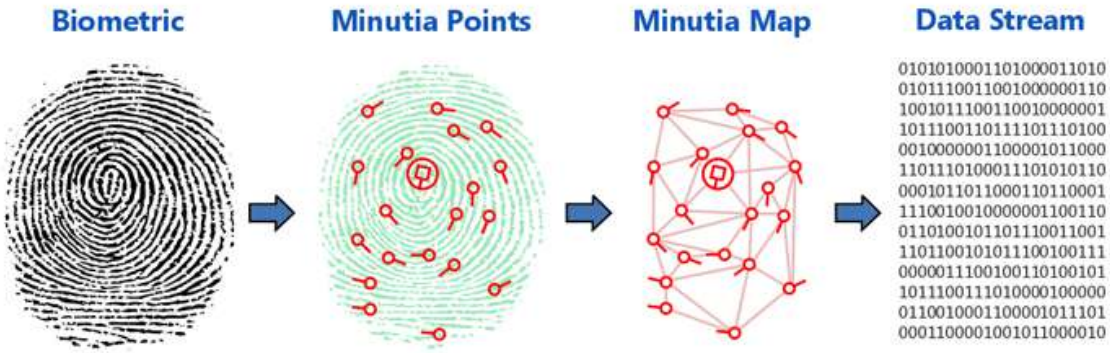


Figura 20 - Esquema del proceso de grabado de una huella por un sistema biométrico [9]

Aunque se han citado algunos de los más importantes, existen otros muchos que también tienen cierta importancia como los sistemas de reconocimiento facial, la ayuda en defensa cibernética o la automatización de procesos robóticos.

3.2 PRINCIPALES EMPRESAS EN EL MERCADO DE LA INTELIGENCIA ARTIFICIAL

Aunque la creación de soluciones de Inteligencia Artificial está siendo una tendencia en los últimos años, creándose gran número de empresas dedicadas a ello, unas pocas privilegiadas dirigen este mercado a una distancia abismal sobre el resto de empresas. Estas grandes compañías, tienen recursos prácticamente ilimitados y llevan años invirtiendo en desarrollo.

Entre estas empresas podemos destacar [10]:

- IBM: es sin duda una de las pioneras en la Inteligencia Artificial ya que sus primeras investigaciones al respecto datan de 1950. El producto más famoso de IBM a día de hoy es Watson, una plataforma basada en aprendizaje automático y procesamiento del lenguaje natural para analizar grandes cantidades de datos no estructurados para sus clientes.
- Google: es una de las empresas que más interés muestran en la Inteligencia Artificial, habiendo invertido millones en el desarrollo y compra de productos. Entre los desarrollos más actuales de Google se encuentran los coches autónomos.

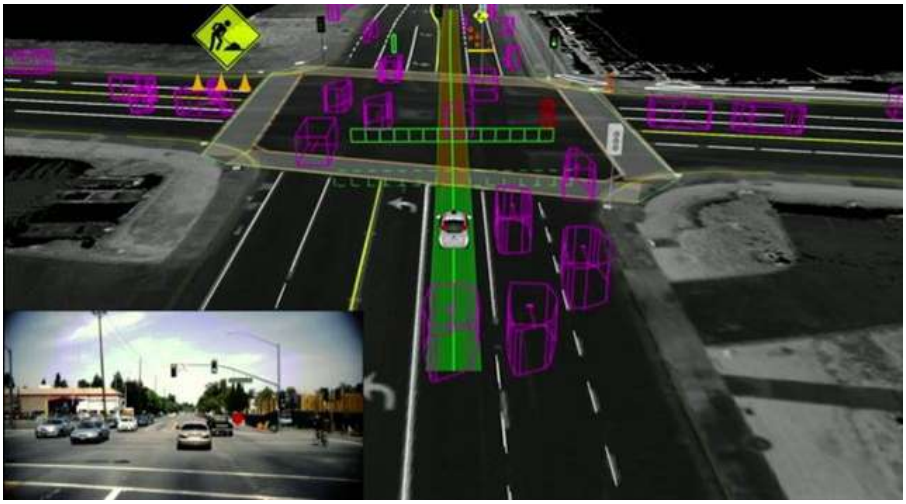


Figura 21 - Visión del mundo por un coche autónomo [11]

- Apple Inc.: aunque es una empresa que vive de los productos electrónicos de consumo, Apple ha invertido grandes cantidades de dinero en el desarrollo de *Siri* como asistente virtual y en técnicas biométricas de reconocimiento facial para el desbloqueo de los dispositivos.
- Facebook: otro de los gigantes tecnológicos también se ha unido al desarrollo de Inteligencia Artificial. Facebook ha confirmado que está construyendo uno de los laboratorios de Inteligencia Artificial más potentes del mundo, porque quiere hacer de la IA el eje de su crecimiento y desarrollo.

3.3 ACCESO A LOS PRODUCTOS DE LA INTELIGENCIA ARTIFICIAL

Entre 2011 y 2014 se invirtieron más de 2 mil millones de dólares en empresas relacionadas con la Inteligencia Artificial [12]. En esos años se vivió una auténtica locura de absorciones y fusiones de pequeñas y medianas empresas dedicadas al desarrollo de soluciones de IA por parte de alguno de los gigantes que se nombraron en el apartado anterior.

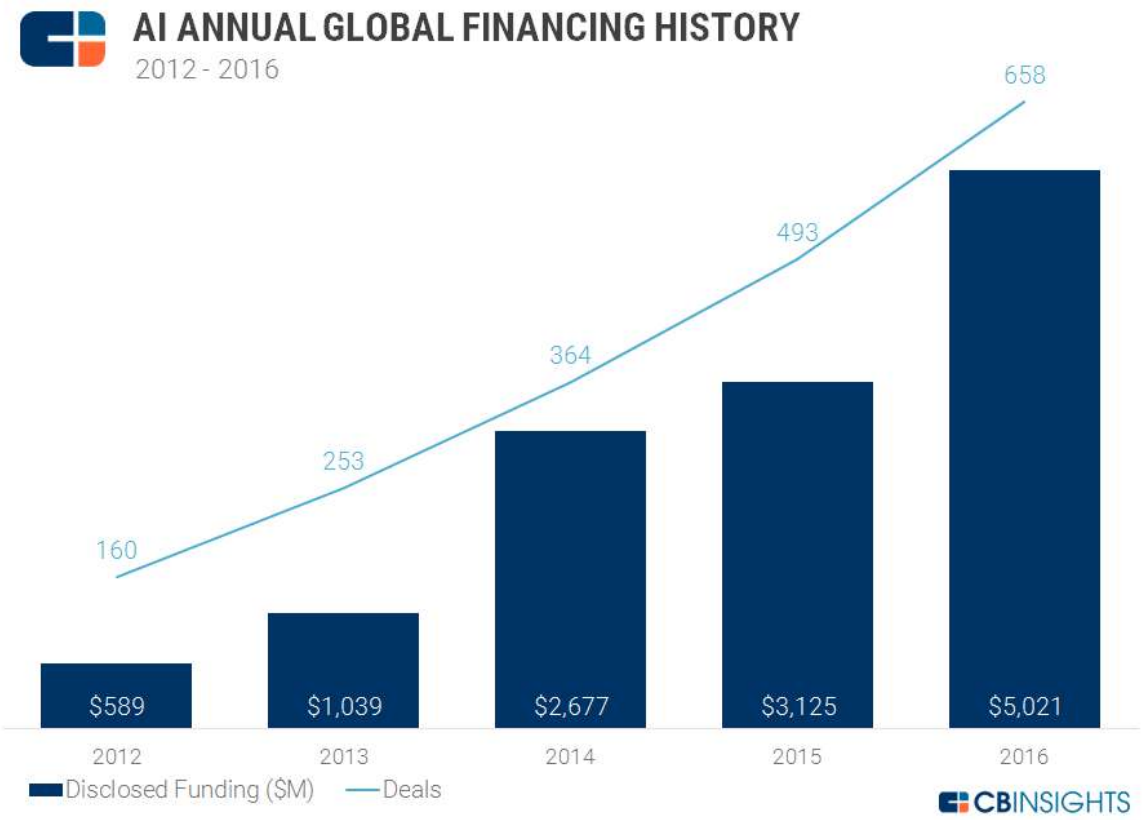


Figura 22 - Evolución de la inversión global en Inteligencia Artificial

No es de extrañar, que tanta inversión haya venido seguida de publicidad sobre las maravillas de los productos ofrecidos. Es por eso, que cada día más, las empresas se conciencian de que la Inteligencia Artificial puede mejorar su eficiencia y aumentar sus beneficios.

Hace unos cuantos años, el tener una página web parecía privilegio reservado a las grandes empresas. Esto ya no es así, ya que la creación de Webs está a la orden del día para negocios de todo tipo y tamaño y a cualquier tipo de coste. Con la Inteligencia Artificial se produce un efecto similar. En un primer momento parece algo futurista y lejano, pero poco a poco se ha ido incorporando a nuestras vidas y llegará un momento en que todo el mundo busque soluciones en los productos de la Inteligencia Artificial.

A día de hoy existen diferentes opciones para poder disfrutar de los beneficios de la Inteligencia Artificial:

- *Outsourcing*: como ya se dijo antes, hay multitud de empresas desarrollando productos de Inteligencia Artificial para generar soluciones a sus clientes. La variedad de productos y empresas donde elegir es tan grande como el rango de precios que puede costar.
- Desarrollo propio: si la empresa tiene un departamento informático de calidad cabe la posibilidad de desarrollar sistemas básicos propios. Para ello existen dos posibilidades principales:
    - Uso de librerías *open-source*: existe gran variedad de fuentes *open-source* que ofrecen algoritmos y modelos para la utilización particular. Uno de los más famosos es la librería de Machine Learning de Google: *TensorFlow*.
    - Desarrollo de algoritmos: dado que muchos algoritmos de la Inteligencia Artificial están publicados en artículos científicos y tienen carácter divulgativo, replicar modelos en software a partir de las fórmulas puede ser una tarea asequible.

CAPÍTULO 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Cuando Fernando Pavón [13] me propuso hace unos meses este proyecto no dudé que podía ser realmente interesante. La Inteligencia Artificial es una tecnología en alza y aunque carecía de base sobre algoritmos o técnicas me parecía un reto al que quería enfrentarme.

La Inteligencia Artificial es la última tendencia dentro de la tecnología. Las mayores inversiones están siendo enfocadas al desarrollo de productos que, si bien pueden no ser solo de Inteligencia Artificial, tienen relación con ella.

El proyecto en cuestión ofrece una solución limpia y eficaz a un problema real al que se enfrentan día a día muchas empresas: la pérdida de clientes. Perder un cliente tiene dos grandes efectos negativos: el primero y más evidente es la consecuencia económica. El segundo, está oculto por debajo de los motivos económicos, y es que hay una razón por la que el cliente no estaba satisfecho que le ha empujado a abandonarme y no he sabido detectarlo.

Una vez se sabe que algoritmos producen mejores resultados es más fácil generar mediante el mismo proceso distintos modelos para la detección de diferentes situaciones. Solamente cambia la situación de negocio y los datos de entrada. De esta forma se consigue que una vez generado el código, el desarrollo sea barato (ya que no se requiere hardware adicional).

El modelo puede ajustarse en función de las necesidades del negocio. Por ejemplo, en el caso que nos ocupa, si una vez hemos detectado que unos de nuestros clientes pueden dejar de serlo, conseguimos invertir la situación ofreciéndole una promoción, es un coste muy bajo a cambio de los beneficios que reporta la fidelización de un cliente. Para este caso podría interesarnos un sistema con una sensibilidad muy alta en la detección de los posibles abandonos, aunque ello conllevara una mayor tasa de falsas detecciones (que como he dicho, tienen un bajo coste).

Sin embargo, otros modelos de negocio como podría ser en el caso de un banco el saber si un cliente es un potencial moroso a la hora de conceder un crédito, pueden preferir una sensibilidad de en torno al 70% con el objetivo de obtener unas tasas de falsos positivos cercanas al 0%.

Se trata de un proyecto que se ha desarrollado con datos reales y aporta una herramienta útil para incrementar los beneficios de una empresa y mejorar la satisfacción del cliente. La detección se realiza con suficiente margen para poder reaccionar.

Aunque el objetivo principal es conseguir que las empresas puedan mantener sus clientes y por tanto sus beneficios, un efecto subyacente de este proyecto es la mejora de la satisfacción general de los clientes, que reciben ofertas y promociones incluso antes de ser plenamente conscientes de que su satisfacción con el servicio recibido estaba yendo a peor.

Para que una detección del abandono tenga sentido, debe hacerse con un margen de reacción para que la empresa pueda intentar invertir la situación. El margen que se ha considerado para hacer la detección es de un mes.

4.2 METODOLOGÍA

Como el proyecto es solamente desarrollo software y tiene unos pasos de desarrollo y prueba muy marcados, se ha seguido una metodología de cascada. Cada bloque del diagrama que se incluye a continuación suponía una pequeña etapa tras la cual se hacía una comprobación con el personal de GAMCO, S.L.

Es una metodología adecuada para este trabajo ya que, al constar de pasos bien marcados, ser realizado por una sola persona, y tener un enfoque y objetivos claros, se ahorra el tiempo en reuniones y papeleo que pueden tener otras metodologías de desarrollo software.

Algunos de los pasos, especialmente en la parte de los algoritmos, constaban de una batería de pruebas para probar diversos parámetros que consumían una gran cantidad de tiempo de computación. Eso hacía que cualquier error en el proceso provocara graves retrasos.

Los objetivos y pasos fueron marcados al principio del proyecto por el Director del mismo y fueron revisados en reuniones periódicas independientemente de la evolución de la metodología de cascada.

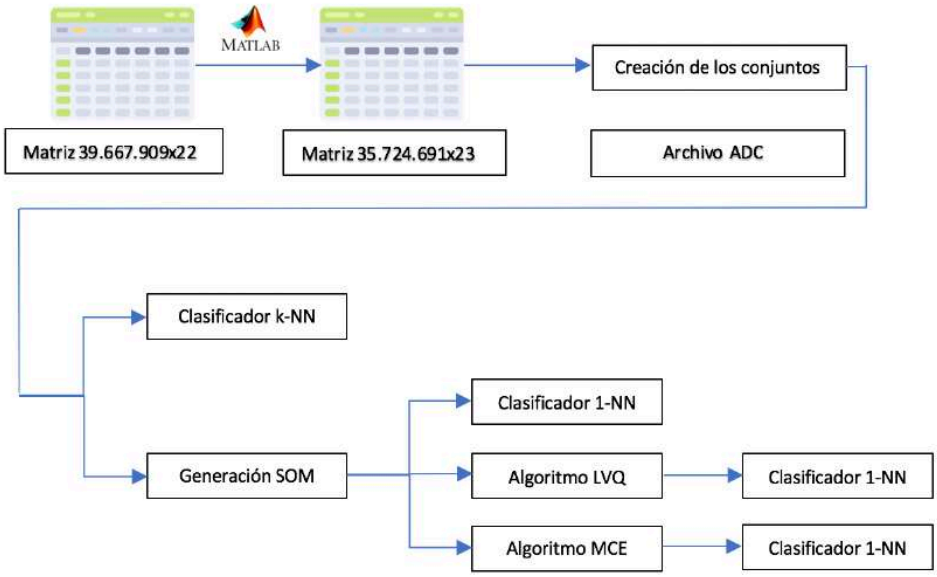


Figura 23 - Diagrama de bloques del proceso seguido en el proyecto

4.3 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

La planificación inicial fue presentada en el ANEXO B. En ella se observa el tardío inicio del proyecto. Además, el proyecto tuvo un retraso considerable debido a errores no detectados en una etapa intermedia del proceso, por lo que finalmente se acabó con casi un mes de retraso respecto a la planificación inicial.

Figura 24 - Cronograma del desarrollo del proyecto

Respecto al coste del proyecto cabe destacar que al haberse desarrollado en las instalaciones de GAMCO, S.L el coste ha sido cero. El ordenador utilizado era propiedad de GAMCO, S.L. La licencia de MATLAB utilizada era la de estudiante proporcionada por la Universidad Pontificia Comillas. GAMCO, S.L contaba además con un ordenador dedicado de alta capacidad para la realización de los cálculos más pesados que podían realizarse en paralelo con MATLAB. El proyecto no requería de ningún tipo de hardware que requiriera inversión.

CAPÍTULO 5. MODELO DESARROLLADO

5.1 SITUACIÓN INICIAL DEL PROYECTO

Inicialmente el proyecto arranca sobre un ordenador en distribución Windows sobre el que lo primero que se hace es instalar una partición del sistema operativo GNU/Linux en su distribución Ubuntu.

Toda la red de GAMCO S.L está montada sobre este sistema operativo por sus ventajas [14]:

- Reducción de costes derivada de que se trata de un sistema operativo *open source* y sus programas son gratuitos.
- Ofrece una gran capacidad de personalización (aunque esto pueda traducirse en complejidad de gestión de la red).
- Es un sistema operativo con mayor eficiencia que Windows: los procesos en Ubuntu consumen menos RAM que en Windows y en términos generales funciona de una forma más rápida y ágil.
- Mayor seguridad: aunque la seguridad depende de la configuración de la red, los sistemas Linux están expuestos a menos ataques y realizan actualizaciones constantemente.

El primer programa que se instalará es MATLAB en su versión 2017b. Se descargarán todos los paquetes para evitar tener que hacerlo después y se utilizará la licencia de estudiante que proporciona la universidad.

Se instalan los diferentes programas y se crean las cuentas en los diferentes repositorios para arrancar el trabajo siguiendo la estructura interna de trabajo y documentación:

- Control de versiones: GitLab es uno de los mejores servicios web de control de versiones. Se abre una cuenta gratuita y se añade el usuario al repositorio creado para el proyecto en cuestión. Posteriormente se clona el proyecto en disco local para mantener el repositorio actualizado. En GitLab se incluyen todos los archivos utilizados en

MATLAB y las Wikis explicando el funcionamiento de los archivos para cada uno de los procesos. Todos los archivos van acompañados de una cabecera en la que se explica el contenido del mismo dependiendo de si se trata de una función o de un *script* de ejecución de código.

Figura 25 - Logo de Gitlab [15]

- Evernote: es una aplicación informática que tiene como objetivo la organización de información mediante el archivo de notas. En Evernote se almacenan resúmenes de las reuniones periódicas para la revisión y análisis e incluso para tenerlas como punto de comprobación de las tareas en reuniones futuras.

Figura 26 - Logo de Evernote [16]

- Subversion: es otra herramienta de control de versiones *open source* basada en un repositorio y cuyo funcionamiento asemeja al de un sistema de ficheros. Subversión, al que normalmente se llama SVN, era el repositorio donde se incluían las funciones no propietarias del proyecto en concreto. Funciones generadas por GAMCO, S.L para otros proyectos y que en algún momento de los scripts que se encuentran en GitLab eran invocadas.
- Repositorio de datos: los datos son uno de los contenidos más sensibles y como tal se tratan con cuidado. Inicialmente los datos estuvieron compartiéndose en la red local

como archivos cifrados. Posteriormente se creó un directorio privado en la red local para compartir todos los archivos de datos sin tener que cifrar y descifrar cada vez que querían utilizarse.

Aunque no lo voy a incluir en la lista anterior ya que no es un programa, es importante destacar que debido a la gran cantidad de datos con la que se hacían los cálculos, fue necesaria la utilización de un ordenador dedicado con alta capacidad de 128 GB de memoria RAM y 18 núcleos para prácticamente todos los pasos.

5.2 PROCESAMIENTO DE LOS DATOS

El proyecto se ha llevado a cabo con datos reales de ventas de una empresa que ya es cliente de GAMCO S.L. Por motivos de confidencialidad no se puede publicar el nombre de la empresa ni datos reales de las ventas de la misma.

El archivo inicial se presentaba como una matriz en formato de archivo de MATLAB (.mat) con unas dimensiones de 39.667.909 filas por 22 columnas. Cada fila pertenecía a una compra por lo que los clientes estaban repetidos a lo largo de la matriz. En total había 118.321 clientes diferentes.

Cabe destacar que los datos pertenecían a un período de aproximadamente cuatro años, por lo que muchos de los clientes que se encontraban en la matriz eran antiguos.

Entre las columnas de la matriz inicial se encuentran: identificadores de los clientes, fecha de los pedidos, fecha de entrega, cantidad de producto, tipos de productos, peso de los mismos, precio, beneficio obtenido por la empresa, descuentos aplicados, tipo de cliente del que se trata, etc.

5.2.1 Definición de los tipos de clientes

Es importante definir claramente que se va a hacer y que se está buscando predecir a partir de los datos iniciales.

El proyecto se llamó "PREDICCIÓN DEL ABANDONO DE CLIENTES", por lo que el evento que tenemos que buscar el momento en que un cliente decide dejar de comprar a nuestra marca.

Dado el período de fechas que teníamos en la matriz inicial y el tipo de negocio del que se trata, definimos que un cliente nos ha abandonado cuando pasa más de 5 meses sin hacer ninguna compra, es decir, 150 días.

Partiendo de este punto podemos poner etiquetas a todos los clientes de la matriz inicial:

- Cliente **ACTIVO**: todo aquel cliente que haya realizado una compra en los últimos cinco meses. Se identificará como un cliente tipo 0.
- Cliente **ABANDONA**: aquel cliente que hace más de cinco meses que no ha realizado ninguna compra y por lo tanto, dejo de considerarlo cliente. Se identificará como un cliente tipo 1, ya que lo que se busca es detectar los abandonos.
- Cliente **RECURRENTE**: este es un caso especial de cliente que después de haber pasado más de cinco meses sin realizar compras, vuelve a hacerlo. Un cliente puede ser recurrente o no recurrente independientemente de que el estado actual del mismo sea activo o inactivo.

Una vez definidos los tipos de clientes, se procede a añadir las etiquetas de los mismos a la matriz inicial de tal forma que ahora las dimensiones de la matriz pasan a ser 39.667.909 filas por 24 columnas, siendo la columna 23 el *flag* que indica si un cliente es actual un 0, o si un cliente ha sido un abandono un 1 y la columna 24 un indicador de si se trata de un cliente recurrente o no (1 para recurrente y 0 para no recurrente).

Por poner cifras a la situación en la matriz inicial, después del etiquetado se tenían:

- 46.954 clientes de tipo ACTIVO.
- 71.367 clientes de tipo INACTIVO.

Nos encontramos por tanto con que aproximadamente el 40% de los clientes son activos.

Analizando los clientes recurrentes vemos que en la matriz solo hay 10.495 clientes de este tipo frente a los 107.826 del tipo NO recurrente.

De acuerdo con los objetivos del proyecto, también es importante dejar claro los parámetros de medición. Para ello se han definido los siguientes conceptos:

- **Positivo Cierto:** aquel cliente que se detecta como potencial abandono y resulta acabar siéndolo. Se abreviará como PC.
- **Positivo Falso:** aquel cliente que se detecta como potencial abandono y no lo era. Se abreviará como PF.
- **Negativo Cierto**: aquel cliente que se considera cliente satisfecho y finalmente es un cliente que no va abandonar a su proveedor. Se abreviará como NC.
- **Negativo Falso:** aquel cliente que el modelo piensa que está satisfecho y acaba siendo un abandono para la empresa. Se abreviará como NF.

Para medir la calidad de los resultados se calculan tres parámetros que se utilizan a lo largo de todo el proyecto:

- **Sensibilidad**: es el número de PC detectados entre la suma de los PC y los NF. Es decir, el porcentaje de clientes que se van que es capaz de detectar.
- **Susceptibilidad:** es el cociente entre los PF y la suma de los PF más los NC. Es decir, el porcentaje de clientes satisfechos que creo el sistema detecta como potenciales bajas.
- **Eficiencia:** el cociente entre PC más NC entre la suma de todos los clientes. Es decir, el porcentaje de aciertos del modelo entre todos los clientes, sumando aciertos en predicción de abandonos y detección de clientes satisfechos.

5.2.2 Eliminación de filas no relevantes

Como se busca predecir el abandono de los clientes y que la empresa tenga capacidad de reacción, no basta con saber en qué momento un cliente va a dejar de serlo. Hay que ser capaz de predecirlo con antelación para poder adelantarse y reaccionar.

De esta forma se ha tomado un criterio a la hora de mantener los registros de las compras de los clientes:

- Si el cliente es de tipo ACTIVO, se mantendrán todas sus compras.
- Si el cliente es de tipo ABANDONO, se cogerán solamente las fechas en las que se consideraba que era plenamente activo, es decir, las fechas anteriores a que su comportamiento comenzara a indicar que podía abandonar. Este período previo al abandono que queremos eliminar se define de dos formas diferentes:
    - Último mes de compras del cliente si este ha realizado más de cuatro compras en el último mes.
    - Últimas cuatro compras del cliente independiente del período en que se realicen si el período entre ellas es de más de un mes.

Después de eliminar las compras consideraras previas al abandono, la dimensión de la matriz de datos es de 36.918.193 filas por 24 columnas. Hemos eliminado por tanto cerca de 3 millones de filas.

El siguiente paso en la purga de los datos iniciales es analizar si realmente compensa incluir a los clientes recurrentes en el proceso de análisis. Los clientes recurrentes, como se ha explicado en la sección de definición, son un caso particular ya que son clientes que han pasado de estado activo a estado abandono por lo menos en una ocasión. Después de analizar estos comportamientos se extraen dos conclusiones:

- Puede que sean clientes temporales: si son clientes que tienen negocios en sitios de temporada como puede ser un chiringuito de playa, puede entenderse que los pedidos se realicen durante unos meses del año coincidiendo con la temporada estival. Sin embargo este mismo hecho hace que no puedan ser considerados como clientes regulares de una empresa y quizá sea mejor no tenerlos en cuenta para el análisis.
- Otra posibilidad es que estos clientes sean aleatorios: clientes que han decidido probar la marca o clientes que por alguna razón prefieren ir rotando entre varias marcas

diferentes. Este hecho también hace que sean poco relevantes para la generación de los modelos.

Se decide por tanto eliminar todos los clientes de tipo recurrente, lo cual es fácil gracias a la etiqueta *flag* incluida en la columna 24 como se indicó anteriormente.

Después de eliminar todos los clientes recurrentes y la columna del *flag* que indicaba esta situación (si ya no hay recurrentes no hay necesidad de indicar los que no lo son, las dimensiones de la matriz son las siguientes): 35.728.967 filas por 23 columnas.

Por último, se decide eliminar también todas aquellas compras asociadas a los productos fuera de catálogo, más por evitar problemas en la generación de conjuntos que por que no fueran relevantes. En cualquier caso, tampoco significaba un gran cambio ya que el total de productos descatalogados era 360. Las dimensiones finales de la matriz después de realizar la eliminación de todo lo que no se consideraba importante es de 35.724.691 filas por 23 columnas. Se han eliminado por tanto poco más de 4.000 filas en relación a estos productos.

Con las dimensiones de esta nueva matriz tenemos:

- 42.338 clientes activos. Hemos perdido aproximadamente el 10% de los clientes activos.
- 62.808 clientes que abandonan. Hemos perdido aproximadamente el 12% de los clientes que abandonan.

El número total de clientes es de 105.146, lo que representa una pérdida de en torno al 11%. No es representativa y además era necesaria para preparar la matriz para la generación de los conjuntos de entrenamiento. Además, la información sobrante podría haber influido en la calidad de los modelos de salida.

5.3 DEFINICIÓN DE LAS CARACTERÍSTICAS DE LOS CONJUNTOS

El siguiente paso en la preparación de los datos para que puedan ser aplicados a los modelos es la definición de las características de los conjuntos. Este proceso genera una matriz a partir de

los datos de los pedidos de cada uno de los clientes. Es decir, habrá una fila en la matriz por cliente que aparezca en la matriz de datos original.

Las características de los clientes se definen en el catálogo de entradas potenciales. Cada una de estas entradas potenciales se centra en uno o varios rasgos de la información de la matriz inicial para generar información relevante para el modelo.

Existen multitud de entradas potenciales que pueden calcularse, las elegidas para este modelo en concreto se incluyen en el archivo CEP que se utiliza para la generación del archivo del que luego se sacarán los conjuntos.

Para que se puedan calcular las características de un cliente, éste debe estar incluido en un archivo que se llama maestro de Clientes y todos los productos de sus pedidos deben estar incluidos en el maestro de Productos. Es por esto que anteriormente decidimos quitar aquellos productos descatalogados, ya que se eliminan del maestro de Productos una vez dejan de estar en vigor.

Entre las entradas potenciales más destacadas están las siguientes:

- cliente: aunque no es una entrada potencial como tal, el código identificador de cada cliente se incluye en la primera columna de la matriz de características para la correcta identificación de los mismos
- sLK: es el sumatorio de la cantidad total de productos entregados a cada cliente en kilogramos y litros
- sCIN: el sumatorio del ingreso neto de la empresa para cada cliente
- sCIB: el sumatorio del ingreso bruto de la empresa para cada cliente
- sBONI: el total de las bonificaciones obtenidas por cada cliente
- sDTOFAC: el total de los descuentos en las facturas de los clientes
- sDTOPP: el total de los descuentos por pronto pago por cliente
- CP: el código postal de cada cliente
- Ce: centro desde que se hace el envío de los productos
- Ubi: ubicación a la que se hace el pedido

- nZact: número de pedidos que el cliente ha realizado
- status: indica un 1 o un 0 dependiendo si se trata de un cliente activo o que abandona

Aunque aquí solo se incluyen algunas de ellas, el archivo final de entradas potenciales tiene una dimensión de 100.848 filas por 144 columnas. En la matriz de entrada había 105.146 clientes por lo que los que faltan no pudieron encontrarse en el maestro de clientes y por tanto no pudo calcularse sus características

Aunque vemos que el archivo tiene 144 columnas, la primera es el código del cliente y la última el tipo de cliente, que no pueden considerarse entradas potenciales como tal.

La estructura que usa el archivo CEP para definir las entradas potenciales que van a utilizarse es la siguiente:

```
<ENTRADAS>
 <NUM_ENTRADAS_SIMPLES>144</NUM_ENTRADAS_SIMPLES>
 <NUM_ENTRADAS_DERIVADAS>0</NUM_ENTRADAS_DERIVADAS>

<SIMPLE>
 <NOMBRE>Cliente</NOMBRE>
 <DESCRIPCION>Indice de clientes</DESCRIPCION>
 <TIPO>20</TIPO>
 <SUBTIPO>1</SUBTIPO>
 <MIN_DESF>0</MIN_DESF>
 <MAX_DESF>0</MAX_DESF>
</SIMPLE>

<SIMPLE>
 <NOMBRE>sLK</NOMBRE>
 <DESCRIPCION>Suma de LK</DESCRIPCION>
 <TIPO>20</TIPO>
 <SUBTIPO>3</SUBTIPO>
 <MIN_DESF>0</MIN_DESF>
 <MAX_DESF>0</MAX_DESF>
</SIMPLE>

<SIMPLE>
```

```
 <NOMBRE>sCIN</NOMBRE>
 <DESCRIPCION>Suma de CIN</DESCRIPCION>
 <TIPO>20</TIPO>
 <SUBTIPO>4</SUBTIPO>
 <MIN_DESF>0</MIN_DESF>
 <MAX_DESF>0</MAX_DESF>
</SIMPLE>
<SIMPLE>
 <NOMBRE>status</NOMBRE>
 <DESCRIPCION>1=activo, 0=inactivo</DESCRIPCION>
 <TIPO>20</TIPO>
 <SUBTIPO>175</SUBTIPO>
 <MIN_DESF>0</MIN_DESF>
 <MAX_DESF>0</MAX_DESF>
</SIMPLE>

</ENTRADAS>
```

Esta estructura con las características de los clientes es la base para que el cálculo de los modelos sea posible. Su generación es lenta pero una vez están hechos facilitan y hacen que el resto sea posible. En el apartado siguiente se hablará de la fragmentación de estos matriz de características para poder realizar correctamente los experimentos.

Después de este paso, se comprueba que los resultados obtenidos para las características de cada cliente son consistentes con lo que existía en la matriz de entrada y en la base de datos de donde se generó la matriz. Se escogen para ello clientes aleatorios y se calculan las características de cada cliente para compararlas con las de la fila que le corresponda a ese cliente dentro de la matriz de características.

5.4 CREACIÓN DE LOS CONJUNTOS

Los modelos que vamos a utilizar van a ser entrenados con datos de las características de los clientes. Sin embargo, una vez se ha producido el entrenamiento, el modelo debe ser capaz de reconocer y clasificar nuevos casos que no haya visto antes, es decir, casos diferentes a los utilizados para el entrenamiento. Es por eso que los clientes deben dividirse en diferentes conjuntos para las distintas partes del proceso.

Los clientes van a dividirse en tres conjuntos:

- **Conjunto de entrenamiento:** se le va a llamar Ce y va a contar con el 70% de las muestras. Será utilizado para entrenar los modelos.
- **Conjunto de pruebas:** se le va a llamar Cp y representará el 20% de las muestras. Será utilizado para verificar la capacidad de clasificación de los modelos después de entrenarlos.
- **Conjunto nuevo:** es el más pequeño de todos y representará un 10% de las muestras. Se utilizará para probar otro conjunto diferente sobre los modelos y poder analizar las conclusiones al final.

Si un modelo está bien generado, deberá ofrecer resultados similares para todos los conjuntos. Si bien es lógico que los resultados para el Ce sean ligeramente mejores que para el Cp, una diferencia notable sería signo de un modelo sobreentrenado. Un modelo sobreentrenado es aquel que por una mala planificación del entrenamiento no es capaz de funcionar correctamente con muestras diferentes a las utilizadas en el proceso de entrenamiento del mismo.

A la hora de planificar los experimentos se decidió crear diferentes conjuntos para probar la robustez de los modelos en función de los resultados. Hay que tener dos cosas en cuenta para la creación de los conjuntos:

- La primera es la distribución de tipos de clientes en el Cx (suma de Ce, Cp, y Cn). Como se mostró en el primer apartado, la cantidad de clientes de tipo activo y tipo abandono no está equilibrada al 50%. Por eso mismo, cabe preguntarse si un modelo entrenado con conjuntos en los que el 60% de los datos no tendrá tendencia a orientar sus clasificaciones a esos porcentajes. De esta forma, se decidió crear 4 conjuntos de datos diferentes equilibrados al 50% entre clientes de los dos tipos. De esta forma, todos los conjuntos incluían a los clientes activos (ya que de ellos hay menor número) y cada conjunto incluía pequeñas variaciones en los clientes que abandonan (no demasiado importantes ya que solo había un 10% más que clientes activos).
- Otro factor a tener en cuenta es un parámetro que se llama "*barajar*". La matriz inicial estaba ordenada cronológicamente y por tanto los clientes que se encuentran tienen

mayor antigüedad y con más posibilidades de ser clientes de tipo abandono. Si no se baraja, alguno de los conjuntos incluirá a todos estos clientes antiguos que contienen más información y otro conjunto tendrá clientes mucho más nuevos (probablemente activos) generando resultados serán confusos.

A continuación se muestra un fragmento del código del archivo ADC con el que se generan los diferentes conjuntos.

```
<?xml version='1.0' encoding='utf-8' ?>
<!--
 <METODO_EXT>: 0 - Secuencial (barajar=false), 1 - Aleatorio
(barajar=true)
-->
<CONJUNTOS>
 <CEP>
  <ID>/CXs/CEP-20-2-imp.xml</ID>
 </CEP>
 <CX>
  <X100CE>70</X100CE>
  <X100CP>20</X100CP>
  <X100CN>10</X100CN>
  <DIMENSION>66</DIMENSION>
  <IDENTRADAS>12</IDENTRADAS>
  <METODO_EXT>0</METODO_EXT>

 <ENTRADA>
  <ID>20@3</ID>
  <DESFASE>0</DESFASE>
  <SALIDA>0</SALIDA>
 </ENTRADA>

 <ENTRADA>
  <ID>20@67</ID>
  <DESFASE>0</DESFASE>
  <SALIDA>0</SALIDA>
 </ENTRADA>

 <ENTRADA>
```

```
 <ID>20@68</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

<ENTRADA>
 <ID>20@69</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

<ENTRADA>
 <ID>20@70</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

<ENTRADA>
 <ID>20@71</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

<ENTRADA>
 <ID>20@72</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

 <ENTRADA>
 <ID>20@73</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>

<ENTRADA>
 <ID>20@4</ID>
 <DESFASE>0</DESFASE>
 <SALIDA>0</SALIDA>
</ENTRADA>
```

```
  <ENTRADA>
  <ID>20@175</ID>
  <DESFASE>0</DESFASE>
  <SALIDA>1</SALIDA>
 </ENTRADA>

</CX>
</CONJUNTOS>
```

Como puede observarse en la tercera línea, el parámetro de barajar se llama *METODO_EXT*.

En las primeras líneas de la definición de los conjuntos se puede ver como se definen los porcentajes que de 70% para el Ce, 20% para el Cp y 10% para el Cn tal y como se indicó anteriormente.

Se puede ver como en el último bloque se indica que la salida es igual a 1. Esto es porque la salida del modelo es el indicador de abandono de los clientes y este estaba incluido en la última columna de la matriz de características de los clientes. De esta forma, cuando se entrenen un modelo, sabrá cuál es la salida que producen los datos que se le están introduciendo.

De los cuatro conjuntos que se crearon para la batería de pruebas (ya se dijo anteriormente que eran cuatro). Dos de ellos se crean sin barajar los conjuntos y dos de ellos barajando. Se podía prever que los conjuntos barajados tendrían mejor resultado y como se mostrará en la sección de resultados, así fue.

5.5 PRUEBAS CON MODELOS

Una vez tenemos los conjuntos creados podemos empezar a las pruebas de los modelos.

Esto es el comienzo de la segunda gran parte del proyecto, y aunque en el fondo sea el objetivo del proyecto, si la etapa de análisis de datos y creación de conjuntos no se hace de forma eficaz, los modelos nunca generarán resultados positivos.

5.5.1 Clasificador k-NN

La primera etapa dentro de las pruebas fue un clasificador k-NN. Como ya se explicó anteriormente es un clasificador simple pero que da buenos resultados. Se realizaron diferentes pruebas con el k-NN.

El objetivo de utilizar tan pronto un clasificador k-NN era comprobar la mejoría que podían tener los resultados después de utilizar modelos más elaborados.

Para la batería de pruebas se tenían las siguientes posibilidades:

- Había que probar los cuatro conjuntos diferentes que se habían creado. Recuerdo que dos de ellos estaban barajados y dos de ellos no.
- Había que elegir el número de vecinos con los que se quería que se compitiera para la clasificación. El modelo más básico es 1 vecino. Se probó además con 3 y 5. No se prueba con vecinos pares ya que en caso de empate se complica la algoritmia de decisión.
- Se prueba por último con diferentes algoritmos de votación:
    - Votación normal: en el que solo se cuentan a que clase pertenecen los elementos más cercanos (será el algoritmo 1).
    - Votación ponderada: en la que se tiene en cuenta a que distancia están cada uno de estos elementos cercanos de la muestra a clasificar para ponderar el valor de su voto (será el algoritmo 2).
- Aunque pueden utilizarse diferentes distancias para la clasificación, para todos los casos se utilizó la distancia Euclídea.

En el proceso para comprobar la calidad de la SOM, el primer paso es cargar la SOM que queramos comprobar y el conjunto de datos con el que se ha realizado

```
fModelo = '/Modelos/SOM_2.mat';
fEntradas = '/Entrena/CXs/CX_4_20xx.mat';
model = load(fModelo); % la som
cx = load(fEntradas); % la cx
```

Guardamos las salidas reales de los conjuntos para poder medir la calidad del modelo una vez se generan las salidas predichas.

```
salidasRealesCe=cx.Ce(:,77);
salidasRealesCp=cx.Cp(:,77);
salidasRealesCn=cx.Cn(:,77);

salidasRealesCe=desNormalizar(salidasRealesCe,
cx.med(1,end),cx.var(1,end));
salidasRealesCp=desNormalizar(salidasRealesCp,
cx.med(1,end),cx.var(1,end));
salidasRealesCn=desNormalizar(salidasRealesCn,
cx.med(1,end),cx.var(1,end));
```

Estas salidas tienen que ser desnormalizadas para que queden como ceros y unos ya que tras la generación de las SOMs, los valores originales se normalizan.

A continuación se muestra el código para generar la estructura que permite a partir de los vectores de los conjuntos de entrenamiento, utilizar el algoritmo k-NN para clasificar los elementos de los conjuntos de prueba

```
patrones = cx.Ce(:,1:end-1);
salidas=cx.Ce(:,end);
salidas=desNormalizar(salidas, cx.med(1,end),cx.var(1,end));
medPat=cx.med(1:end-1);
varPat=cx.var(1:end-1);
BC =
struct('prot',patrones,'salProt',salidas,'med',medPat,'var',va
rPat);
```

A continuación se muestran los valores que tiene los parámetros para que el clasificador sea 1-NN. Al ser un algoritmo 1-NN no sería relevante el tipo de algoritmo utilizado ya que se ponderen o no los votos de los vecinos, al haber solo un voto, el del vecino más cercano, este será siempre el ganador. Para el resto de valores de k utilizados (3 y 5) el algoritmo sí que es relevante y se ha probado tanto con el algoritmo 1 como con el 2 (ya se explicaron las diferencias entre ambos anteriormente).

```
nv =1; % número de vecinos
algoritmo = 2;
```

Una vez están introducidos todos los parámetros, podemos calcular las salidas predichas para el Cp. Para ello se itera sobre cada elemento del conjunto de pruebas y se va creando una estructura que se evalúa con el clasificador k-NN que se ha generado a partir del

```
predichoCp=[];
for i=1:length(cx.Cp)
  caso=cx.Cp(i,1:end-1);
  Caso =   struct('Cx',caso,'indicesCx',NaN,'salCx',
[],'med',medPat,'var',varPat);
  [result_kNN,~,~,protCaso,distProtCaso] = evaluate_kNN
(nv,thetasPat,deltakNN,algoritmo,forcePred,BC,Caso,fReskNN,deb
ugEnable);
  predichoCp(i,1)=result_kNN(1,2);
end
```

Se repite el proceso anterior para el Cn:

```
predichoCn=[];
for i=1:length(cx.Cn)
  disp(i);
  caso=cx.Cn(i,1:end-1);
  Caso =
struct('Cx',caso,'indicesCx',NaN,'salCx',[],'med',medPat,'var'
,varPat);
  [result_kNN,~,~,protCaso,distProtCaso] = evaluate_kNN
(nv,thetasPat,deltakNN,algoritmo,forcePred,BC,Caso,fReskNN,deb
ugEnable);
  predichoCn(i,1)=result_kNN(1,2);
end
```

Estos vectores de valores predichos se comparan con los vectores de salida originales Aunque los resultados completos se explicarán en el apartado correspondiente, a continuación se incluye una muestra de las características de los mejores modelos:

Nº prueba	k-NN	Distancia	Sensibilidad	Susceptibilidad	Eficacia
18	5	Ponderado	0.76	0.28	0.74
19	1	Normal	0.72	0.32	0.70
23	5	Normal	0.77	0.29	0.74

Tabla 1 - Resultados para el clasificador k-NN

Puede observarse que los mejores resultados son para los conjuntos barajados como se había previsto. Gracias a esta comprobación a partir de ahora se utilizarán solamente los conjuntos barajados para las pruebas siguientes ahorrando mucho tiempo de computación ya que sabemos que los resultados de los otros conjuntos serán peores.

Hay que destacar que, aunque los modelos presentan pequeñas diferencias, estas no nos hacen poder elegir unos parámetros concretos como los mejores para este clasificador. Lo único que podemos afirmar es que los conjuntos barajados ofrecen mejores resultados.

5.5.2 Generación de las SOM

El siguiente paso fue generar las SOM para los conjuntos que ya sabíamos que proporcionaban los mejores resultados: los dos conjuntos barajados.

Primero había que definir los parámetros la topología de la SOM:

```
% Topología
dimX = 20;
dimY = 20;
posicionTipo = 'HEXTOP'; %GENM = 2
distanciaTipo = 'EUCLDIST'; %GENM = 4
```

Se define una red neuronal de 20 x 20, es decir, 400 neuronas. Se declara que la distancia utilizada para el cálculo de las intensidades y el vector de pesos será Euclídea. Se ha utilizado la distancia Euclídea para medir todas las distancias durante el proyecto.

Una vez se tienen los parámetros se crea la topología de la SOM:

```
topologia = crearTopologia
(dimX,dimY,[],posicionTipo,distanciaTipo);
```

A continuación se definen los parámetros de entrenamiento de la SOM:

```
% Parámetros de entrenamiento
alpha_dec = [0.35 0.015];
sigma_dec = [max(max(topologia.distancias)) , 0.65];
tau = 5.5;
npasadas = 50;
asimetrico = 0; %0 -> usar la salida para el cálculo de las
distancias.
```

El significado de cada uno de los parámetros arriba indicados es el siguiente:

- Alpha_dec: vector del paso de entrenamiento y su decremento.
- Sigma_dec: vector con la ventana de vecindad y su decremento.
- Tau: parámetro para la modificación del paso de entrenamiento (alpha) según la distancia del nodo al nodo ganador. Tomará valores entre 0 y 1.
- Npasadas: número de pasadas de entrenamiento. El número de veces que se pasan los vectores de entrenamiento.
- Asimetría: si asimetría es uno, los vectores de entrenamiento no conocen la salida. Si asimetría es igual a cero, el modelo se entrena conociendo la salida del los vectores de entrenamiento.

Cuando tenemos todo definido, se crea la estructura de entrenamiento que se pasará la función que se encarga de generar la SOM.

```
%% Crear estructura de entrenamiento
parEntrena =
struct('alpha_dec',alpha_dec,'sigma_dec',sigma_dec,'tau',tau,'
npasadas',npasadas);

%% Entrenamiento
```

```
  [w, cuant_e, cont, patClase, fCtes, fRes] = ...
    en3naSOM
(fCx,topologia,parEntrena,asimetrico,idCx,idModelo,pathRes);
```

Uno de los parámetros que se definió arriba es la simetría. Este parámetro hace referencia a si durante el entrenamiento la SOM recibe las salidas del modelo. En este caso, como las incluimos en la matriz de características de los clientes, la SOM está recibiendo las salidas por tanto, al poner asimétrico a 0, estamos diciendo que nuestra SOM será simétrica y utilizará la salida para calcular los vectores de pesos de cada neurona.

A continuación se muestra la diferencia de clasificación de clientes de la SOM para el mismo conjunto (barajado) en función de si se genera la SOM asimétrica (sin salida) o simétrica (con salida):

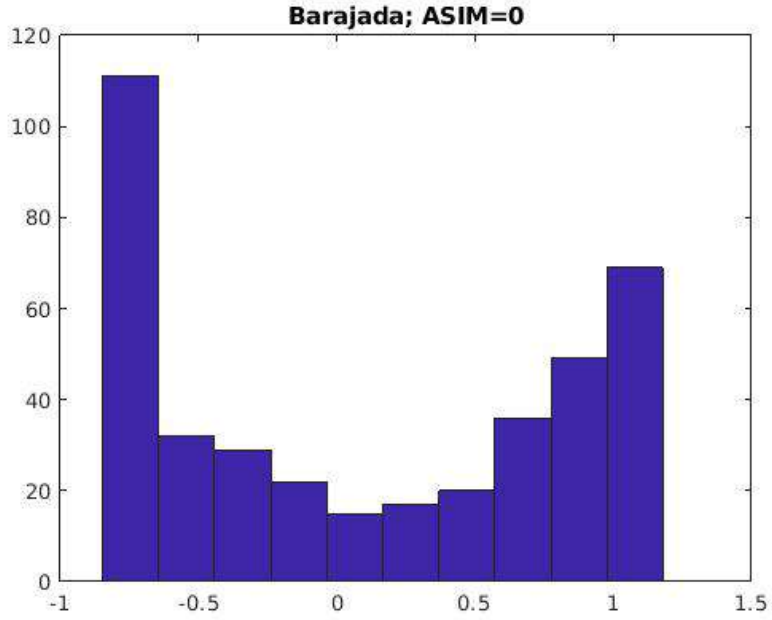


Figura 27 - Distribución de los tipos de clientes en la SOM asimétrica

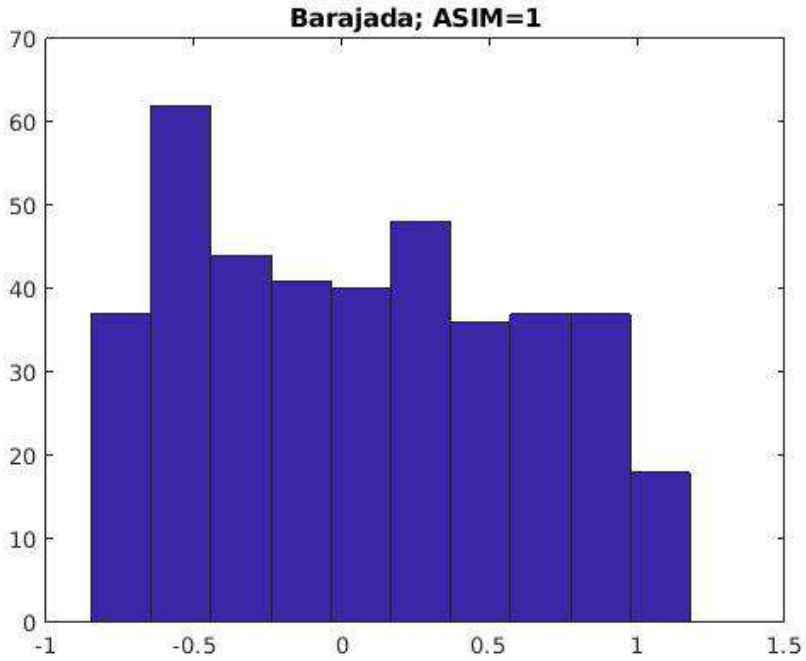


Figura 28 - Distribución de los tipos de clientes en la SOM simétrica

La diferencia es evidente, cuando se entrena utilizando la salida, el modelo clasifica la mayoría de clientes de forma clara, la mayoría de clientes están colocados en los extremos, es decir, el modelo lo está distinguiendo como miembros de un grupo o de otro con exactitud.

Cuando observamos el segundo gráfico vemos que la distribución de clientes es mucho más uniforme. Aunque no es un mal resultado (un mal resultado sería una alta concentración en el centro y una baja concentración en los extremos), esta distribución uniforme dice que no hay perfiles concretos que distingan las dos clases que queremos clasificar.

Los parámetros que se probaron para la generación de las SOMs fueron los siguientes:

```
cx = [4;5;6;7];
alpha = [0.35/2;0.35;0.35*2];
tau=[5.5/2;5.5;5.5*2];
npasadas=[25;50;75];
asimetrico=[0;1];
```

Para comprobar la calidad de las más de 130 SOMs que se generaron, se utilizó un clasificador 1-NN para analizar los resultados que se obtenía al evaluar los Cp. No se incluye el código del algoritmo 1-NN porque es igual al que se utilizó en el apartado anterior para el caso de k=1 (1 vecino)

Los resultados fueron un poco peores de lo que se podía esperar. Los mejores resultados están resumidos en la siguiente tabla:

Nº SOM	Error cuantif.	Sensibilidad	Susceptibilidad	Eficacia
125	4.04	0.84	0.41	0.71
127	4.01	0.83	0.39	0.72
128	3.95	0.86	0.45	0.70

Tabla 2 - Resultados para la clasificación 1-NN de las SOM

5.5.3 Algoritmo LVQ

El algoritmo de aprendizaje de cuantificación vectorial se llevó a cabo justo a continuación. Una vez se habían generado las SOMs el proceso era más rápido a nivel de programación ya que las SOM pueden utilizarse directamente en el algoritmo y el clasificador 1-NN para la evaluación del resultado también estaba preparado

La amplia batería de pruebas que se realizó obligó a paralelizar la realización de las mismas en un proceso que duró varias horas.

En el código que se incluye a continuación se ve como se carga la SOM que queremos utilizar para calcular el VLQ. En este caso y para reducir el número experimentos, utilizamos la SOM 128, que fue la que dio el mejor resultado en el clasificador 1-NN para todas las SOMs.

```
fSOM = '/Modelos/SOM/SOM_salDesN/SOM_128.mat';
som=load(fSOM);
```

Primero guardamos la ruta en una variable y luego se carga. Esto es útil porque en el archivo de resultados, si guardamos la variable, estaremos incluyendo la ruta de la SOM que se ha utilizado por si quisiera comprobarse algo.

Para este algoritmo también necesitamos conocer y tener el conjunto de datos que se utilizó para generar la SOM ya que de ahí sacaremos la media y la varianza que utilizaremos para desnormalizar la información. Como se puede ver en el código incluido a continuación, el código del conjunto se incluye en la ruta del mismo en una variable guardada dentro de la SOM.

```
numeroCX_string=som.fCE(end-9);
num_cx_int=str2num(numeroCX_string);
fCE=sprintf('/LVQ/CX/CX_%d_20xx.mat',num_cx_int);
conjunto_utilizado=load(fCE);
```

El siguiente paso será definir los parámetros que queremos evaluar para el experimento con el algoritmo LVQ:

```
vect_alpha=[0.001; 0.011541746291507; 0.1];
vect_dec_alpha=[0.90 ;0.95 ;0.99];
vec_nPasadas=[25 ;50; 75 ];
vec_win=[0.8 ;0.9 ;0.99];
```

Podemos ver que se prueban tres valores diferentes por parámetro, esto generará un total de 81 modelos. Para generar todas las combinaciones se utiliza una función que crea una matriz en la que se incluye en cada fila una combinación diferente incluyendo los parámetros en las columnas:

```
parametros = {vect_alpha,vect_dec_alpha,vec_nPasadas,vec_win}
[variaciones]=creaVariaciones(parametros);
```

La variable variaciones se utilizará para iterar en un bucle y cambiar los parámetros de entrada al modelo LVQ que se estará aplicando sobre la misma SOM.

Los tres valores que se prueban para cada parámetro se guardan en un vector que se pasa al generador de la matriz de variaciones. Los parámetros que se varían los siguientes:

- Alpha: paso de entrenamiento del algoritmo LVQ.
- Dec_alpha: decremento de alpha en las iteraciones.
- nPasadas: número de pasadas por los patrones de entrenamiento.

- Win: ventana para la adecuación de los prototipos según el algoritmo LVQ.

Como ya se dijo antes y se puede comprobar en el código que se incluye a continuación, cada una de estas variaciones es probada a través de un bucle *for* (que no se incluye porque no es relevante). Cada fila que se saca de la matriz es incorporada a las variables que se introducen al modelo. Como se dijo antes, la media y la varianza necesarias para desnormalizar se sacan del conjunto utilizado.

```
fila_parametros=variaciones(i,:);

med=conjunto_utilizado.med;
var=conjunto_utilizado.var;

alpha=fila_parametros(1);
dec_alpha=fila_parametros(2);
nPasadas=fila_parametros(3);
win=fila_parametros(4);
gamma=[];
```

Por último, una vez tenemos toda la información necesaria, invocamos al modelo. Se realizarán todas las iteraciones que indique el parámetro correspondiente y se guardará el modelo en la ruta que se le pasa por la variable *fRes* (cuyo valor tampoco se ha incluido ya que no es esencial).

```
[Prot salProt] = trainLVQ_SOM_mv
(alpha,dec_alpha,gamma,nPasadas,win,fCE,med,var,fSOM,fRes)
```

Una vez se generaron todos los archivos LVQ, se procede a clasificarlos. De nuevo, no se ha incluido el código porque el clasificador k-NN se utilizó anteriormente y para este caso se utilizaba la versión más básica en la que solo se tomaba en cuenta el vecino más cercano.

Aunque los resultados se verán con detalle en el apartado siguiente, a continuación se incluye un resumen de los mejores modelos:

Nº LVQ	Sensibilidad	Susceptibilidad	Eficacia
19	0.81	0.28	0.76
22	0.80	0.27	0.76
28	0.82	0.28	0.77

Tabla 3 - Resultados para la clasificación 1-NN del algoritmo LVQ

Ahora si se nota una mejoría respecto al clasificador k-NN inicial y por supuesto respecto al resultado obtenido de clasificar la SOM.

5.5.4 Algoritmo MCE

Por último había que probar el algoritmo MCE. Como su nombre indica, el algoritmo de clasificación de los errores mínimos (*minimum classification error* en inglés), intenta reducir el error introducido en la generación de las fronteras de la SOM, moviendo ligeramente las fronteras entre neuronas.

El proceso para generar los resultados es muy similar al seguido con el algoritmo LVQ. Primero se guarda la ruta de la SOM que se quiere utilizar y del conjunto que se utilizó para su creación:

```
fEntradas='/Entrena/CXs/CX_6_20xx.mat';
fSOM ='/Modelos/SOM/SOM_salDesN/SOM_128.mat';
```

Para este algoritmo no hace falta cargar ninguno de los dos archivos ya que eso lo realiza la función internamente.

A continuación se genera la matriz de vectores de prueba de los parámetros que vamos a modificar para intentar encontrar el modelo óptimo:

```
% Variaciones de parámetros para MCE
alpha = [0.1; 0.0115417463; 0.001];
decAlpha = [0.99; 0.95; 0.8];
Psi = [0.75;0.965;0.990];
incPsi = [1.1;1.3;1.5];
nPasadas = [25;50;75];
lambda=[] ;
```

El proceso es exactamente igual que antes: se guardan los valores que queramos probar en vectores que posteriormente se hacen una estructura de vectores que se pasará a la función *creaVariaciones* para que nos devuelva la matriz con todas las combinaciones posibles de los parámetros ordenadas en filas.

Los parámetros que vamos a variar son los siguientes:

- Alpha: paso de entrenamiento del algoritmo MCE.
- decAlpha: ritmo de decremento de alpha.
- Psi: parámetro dinámico de modificación del paso de entrenamiento. También se usa para el cálculo del *"missclasification loss"*.
- incPsi: incremento del Psi.
- nPasadas: número de iteraciones por los patrones de entrenamiento.

```
parametrosNombres =
{'alpha','decAlpha','Psi','incPsi','nPasadas'};
parametros = {alpha,decAlpha,Psi,incPsi,nPasadas};

%% Cálculo de experimentos
[variaciones]=creaVariaciones(parametros);
```

A continuación se observa como en la misma línea que se siguió anteriormente, cada fila de la matriz de variaciones de los parámetros utilizada para generar diferentes modelos a través de una iteración en un bucle *for* (que de nuevo se decide no incluir por no ser relevante).

```
alpha  = variaciones(j,1);
decAlpha = variaciones(j,2);
pSi   = variaciones(j,3);
incPSi  = variaciones(j,4);
nPasadas  = variaciones(j,5);
```

Por último, una vez hemos guardado todos valores en las variables correctas, se invoca al algoritmo MCE. El resultado se guardará en fRes.

```
[patrones salidas] = trainMCE_SOM
(alpha,decAlpha,pSi,incPSi,lambda,nPasadas,fEntradas,fSOM,fRes
);
```

Una vez se haya completado el proceso, se cogerán todos los archivos generados (uno por modelo con todos los diferentes parámetros de entrada) y se pasarán por el clasificador que hemos utilizado en el resto de ocasiones: el 1-NN. De esta forma, podemos observar las diferencias reales entre los algoritmos probados ya que el clasificador utilizado para comprobar los resultados es el mismo para todos.

A continuación se incluye un breve resumen de los resultados que se ampliarán en el siguiente apartado. Son ligeramente mejores que para LVQ, y sobre todo destaca una menor tasa de falsos positivos.

Nº MCE	Sensibilidad	Susceptibilidad	Eficacia
66	0.80	0.23	0.77
71	0.81	0.23	0.76
72	0.81	0.24	0.76

Tabla 4 - Resultados para la clasificación 1-NN del algoritmo MCE

CAPÍTULO 6. ANÁLISIS DE RESULTADOS

Una vez realizado todo el proceso y comprobado que los resultados son correctos se procede a hacer análisis de los resultados. El objetivo del proyecto era encontrar un modelo que se ajustara bien a la situación que se nos planteaba, detectar el abandono de clientes de una empresa. También se buscaba medir las diferencias entre los modelos que se iban probando así como la mejora que se obtenía según aumentaba la complejidad de los mismos.

Se han generado muchísimos modelos, utilizando diferentes algoritmos y haciendo baterías de pruebas variando los parámetros. Como se esperaba inicialmente, los últimos resultados de los algoritmos más complejos fueron los mejores.

6.1 CLASIFICADOR K-NN

Una vez realizado el proceso de limpieza de la matriz de datos y la creación de los conjuntos, se empezó con el clasificador k-NN. Es un algoritmo bastante utilizado ya que sin ser demasiado complejo, ya que la idea de los vecinos más cercanos sobre la que se fundamente es simple, ofrece buenos resultados.

El mayor inconveniente de realizar esta clasificación directamente sobre los conjuntos es el tiempo de computación ya que la clasificación se realiza fila a fila, cliente a cliente. Si se hiciera a través de una SOM, el proceso se agiliza considerablemente. Por lo demás los resultados obtenidos fueron bastante positivos.

Como se realizaron múltiples experimentos se obtuvieron muchos resultados, la tabla con todos se incluye a continuación. Posteriormente se añadirá un resumen de la misma que ya se ha mostrado anteriormente.

Nº prueba	k-NN	CX	Distancia	Sensibilidad	Susceptibilidad	Eficacia
1	1	No barajado	Normal	0.65	0.44	0.61
2	1	No barajado	Ponderado	0.65	0.44	0.61
3	3	No barajado	Normal	0.68	0.45	0.60
4	3	No barajado	Ponderado	0.68	0.44	0.60
5	5	No barajado	Normal	0.70	0.46	0.61
6	5	No barajado	Ponderado	0.70	0.46	0.61
7	1	No barajado	Normal	0.65	0.44	0.60
8	1	No barajado	Ponderado	0.65	0.45	0.60
9	3	No barajado	Normal	0.68	0.45	0.61
10	3	No barajado	Ponderado	0.68	0.46	0.61
11	5	No barajado	Normal	0.70	0.45	0.62
12	5	No barajado	Ponderado	0.73	0.46	0.62
13	1	Barajado	Normal	0.73	0.32	0.70
14	1	Barajado	Ponderado	0.75	0.32	0.70
15	3	Barajado	Normal	0.75	0.30	0.73
16	3	Barajado	Ponderado	0.76	0.30	0.73
17	5	Barajado	Normal	0.77	0.28	0.74
18	5	Barajado	Ponderado	0.76	0.28	0.74
19	1	Barajado	Normal	0.72	0.32	0.70
20	1	Barajado	Ponderado	0.72	0.32	0.70
21	3	Barajado	Normal	0.75	0.30	0.73
22	3	Barajado	Ponderado	0.75	0.30	0.73
23	5	Barajado	Normal	0.77	0.29	0.74
24	5	Barajado	Ponderado	0.77	0.29	0.74

Tabla 5 - Resultados para la clasificación k-NN

Como se dijo en el apartado del desarrollo de este clasificador, los resultados que ofrecieron mejores resultados fueron los de los conjuntos barajados, al elegir aleatoriamente el orden de los clientes que formaban los conjuntos, el entrenamiento del mismo era más eficaz.

Como resumen se adjunta la tabla con los mejores resultados.

Nº prueba	k-NN	Distancia	Sensibilidad	Susceptibilidad	Eficacia
18	5	Ponderado	0.76	0.28	0.74
19	1	Normal	0.72	0.32	0.70
23	5	Normal	0.77	0.29	0.74

Tabla 6 - Mejores resultados para la clasificación k-NN

La sensibilidad hace referencia al porcentaje de clientes que abandonan que se detectan. En el mejor de los casos para la prueba 23, se detectan el 77% de los clientes que pueden dejar de ser clientes. La susceptibilidad hace referencia al porcentaje de falsos positivos dentro de los clientes señalados como posibles abandonos.

6.2 RESULTADOS SOM Y SOM + 1-NN

La generación de los mapas auto-organizados fue un paso importante. Las SOMs son muy útiles por varias razones. En primer lugar, al generarse un mapa de distribución de clientes, hace que la clasificación con k-NN será mucho más rápida y al ser una clasificación neuronal, con los vectores de pesos permite utilizar los algoritmos LVQ y MCE.

Las SOMs también permiten ser representadas gráficamente, generando gráficos y mapas en dos y tres dimensiones que ayudan a ver la distribución de las clases en las neuronas e intuir la calidad del modelo.

La generación de las SOMs fue un proceso largo. Una vez se acabó el proceso de generación, pudo verse cuál de ellas tenía mejor calidad para ser utilizada en los pasos siguientes con el algoritmo LVQ y MCE. Los resultados de las SOMs fueron ligeramente peor de lo que se esperaba.

En la siguiente imagen, se encuentra la distribución de los porcentajes de tipos de clientes por cada neurona. Podemos ver claramente como en la imagen de la izquierda, los clientes activos se concentran en las neuronas de la esquina superior derecha. Además, en esta misma imagen y siguiendo la leyenda de calor situada a la derecha, la parte del mapa con menos concentración

de clientes activos por la derecha el lateral derecho, justo debajo de la zona de máxima concentración de clientes activos.

Si ahora miramos a la imagen de la derecha vemos exactamente el mapa complementario. Estamos observando los clientes que abandonan (inactivos). La parte de más baja concentración porcentual en las neuronas es la superior derecha, coincidiendo y complementando al mapa de distribución de los clientes activos. Y la parte de mayor concentración de clientes que abandonan también coincide.

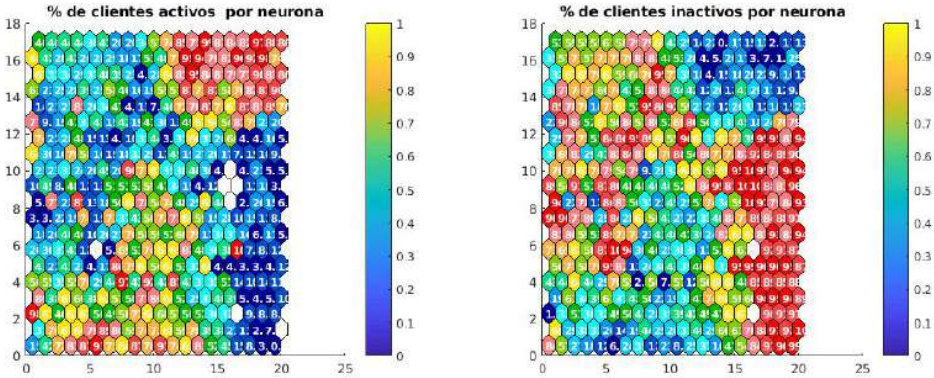


Figura 29 - Mapa auto-organizado de la distribución de los clientes por neurona

Es positivo que la SOM agrupe en zonas claramente diferenciadas altas cantidades de instancias de cada tipo, sin embargo, esto es a nivel porcentual, por lo que hay que comprobar la cantidad de clientes que hay en cada neurona, es decir, aunque el porcentaje de concentración de clientes activos en la esquina superior derecha sea de un 90%, el modelo no tiene por qué ser bueno, ya que puede que en esa neurona se encuentren clasificados 10 clientes, mientras que en una neurona de la zona neutra (que no está claramente orientada a ninguna clase) haya neuronas con 100 clientes activos y 100 inactivos, teniendo por tanto un peso del 50% y siendo mucho más representativas.

A continuación se va a mostrar una vista en tres dimensiones de los mapas que acabamos de describir. Podrá observarse que la concentración de porcentajes de clientes inactivos es más representativa en el mapa que la de clientes activos, que no tiene una zona de elevación clara y tiene muchos puntos de elevación en la superficie del mapa.

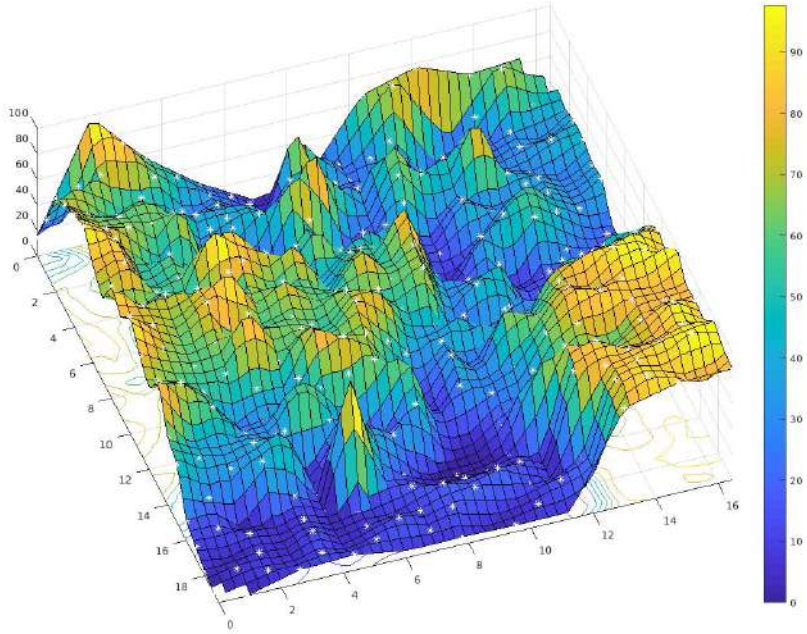


Figura 30 - Visualización en 3D de la SOM de los clientes activos

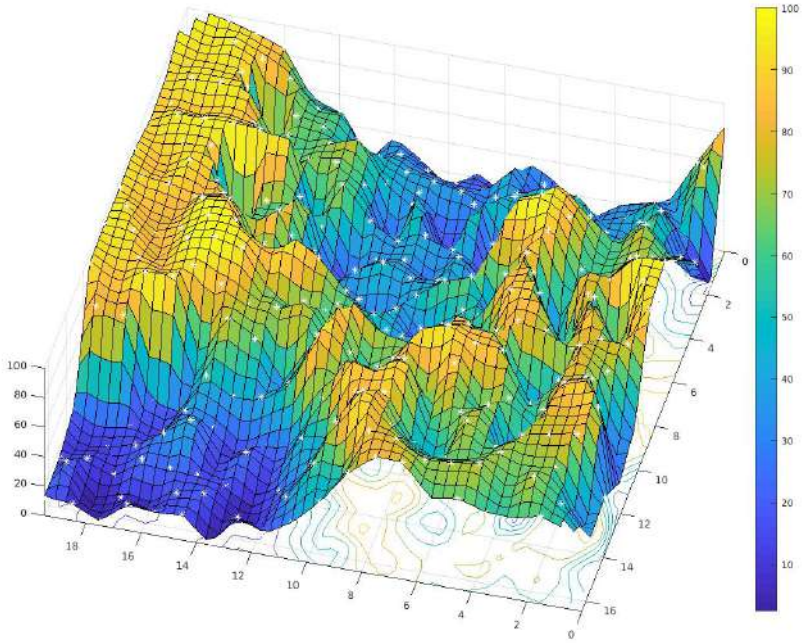


Figura 31 - Visualización en 3D de la SOM de los clientes inactivos

Aunque ambos mapas están en orientaciones diferentes se puede observar que son complementarios. La parte elevada de la imagen inferior, representa las zonas de concentración de clientes que abandonan, esta misma elevación se traduce en una zona hundida de baja concentración de este tipo de clientes en la imagen superior. También podemos observar que la parte que corresponde a la elevación por alta concentración de clientes activos en la imagen superior, se traduce en una zona hundida en la esquina de la imagen inferior.

A continuación van a mostrarse el número de clientes de cada tipo que se encuentran en las neuronas. En la izquierda observamos como no hay una zona definida en tonos amarillo y rojo en la que haya mayor número de clientes activos como podría intuirse cuando vimos la concentración porcentual de tipos en cada neurona. Sin embargo, sí que podemos destacar que la parte con menos clientes activos corresponde a la zona de la SOM en la que se concentran mayor porcentaje de clientes de abandonan.

Aunque a nivel porcentual haya zonas de clientes activos bien definidas, esta baja concentración de clientes activos en una sección del mapa es lo que provoca que en la representación de la SOM en tres dimensiones no hayamos podido distinguir una zona de elevación más concreta como ha sucedido para los clientes que abandonan.

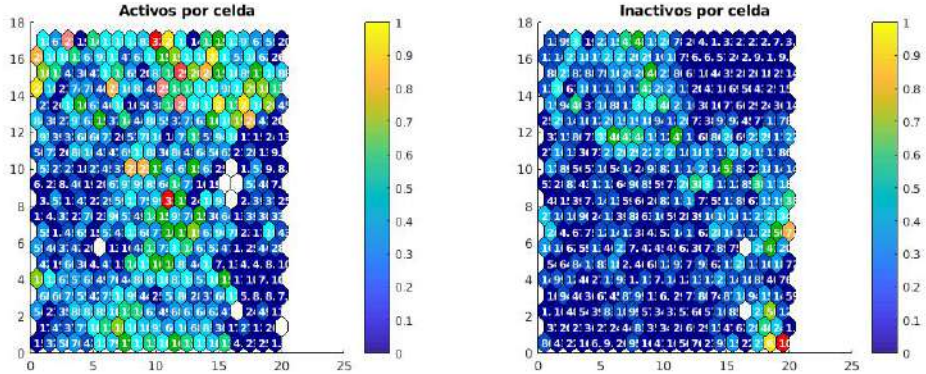


Figura 32 - Distribución del número total de clientes de cada tipo por celda

En la imagen de la derecha se incluye el total de clientes que abandonan por celda. Ahora sí que podemos destacar que la mayoría del mapa tiene un color azul oscuro que indica bajo número de clientes. Esto es más positivo que el caso anterior, ya que provoca que los clientes

de tipo inactivo estén más concentrados (cosa que sucede tanto en número total de clientes como en porcentaje). Las zonas con mayor concentración de clientes inactivos mantienen una relación bastante fiel a la distribución porcentual de clientes tipo abandonan que vimos un poco más arriba. También cabe destacar una neurona en concreto, en la esquina inferior derecha del mapa de clientes que abandonan, que es la zona de mayor concentración de clientes de este tipo y única neurona del mapa con esta alta concentración.

Para acabar de aclarar esta situación se van a añadir dos imágenes más en las que se va a ver por separado, el total de clientes activos y la concentración porcentual de los mismos y por otro lado el total de clientes que abandonan y la concentración de los mismos por neurona.

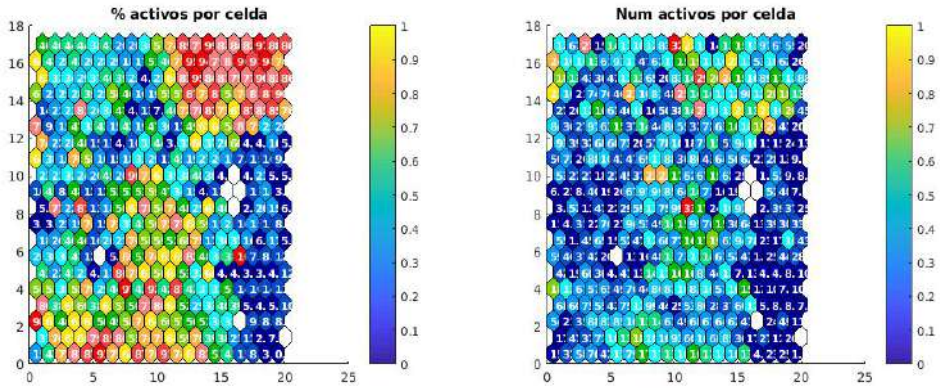


Figura 33 - Distribución porcentual y total de clientes activos por neurona

Como se vio anteriormente, el porcentaje de clientes activos por celda tiene una zona de alta concentración bien definida en la esquina superior derecha y una zona de baja concentración en la parte inferior derecha. A la derecha podemos ver como se mantiene la baja concentración en la parte inferior derecha (que se corresponde además con el bajo porcentaje de abandonos), sin embargo no podemos encontrar una zona de alta concentración de clientes activos que debería haberse encontrado en la esquina superior derecha. Los clientes de tipo activo están distribuidos por diferentes zonas del mapa.

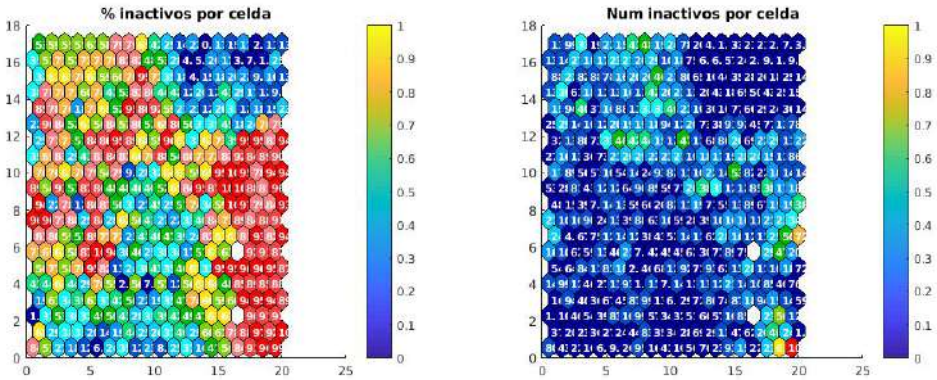


Figura 34 - Distribución porcentual y total de clientes inactivos por neurona

En el caso contrario, los clientes inactivos tienen una distribución porcentual que sí que coincide con la concentración de clientes de cada tipo en la superficie del mapa.

Ya se ha dicho que el resultado de las SOM fue ligeramente peor al que se esperaba. Las imágenes que acaban de explicarse corresponden a la SOM que dio mejores resultados, la 128.

Con las SOM generadas, los resultados para el clasificador 1-NN fueron los siguientes:

Nº SOM	alpha	nPas	tau	Asimetr.	Error cuantif	Sensibilidad	Susceptibilidad	Eficacia
125	0.35	50	2.75	1	4.04	0.84	0.41	0.71
127	0.35	75	5.5	0	4.01	0.83	0.39	0.72
128	0.35	50	2.75	1	3.95	0.86	0.45	0.70

Tabla 7 - Mejores resultados para la clasificación 1-NN de las SOM

6.3 RESULTADOS LVQ + 1-NN

Para el algoritmo LVQ se esperaba que el salto de calidad fuera considerable. Si comparamos los resultados que se obtuvieron con los clasificador k-NN, la diferencia no es demasiado grande. Sin embargo, si analizamos los resultados que se habían generado al clasificar la SOM la diferencia es abismal. Cuando se vieron los resultados de la SOM se llegó a pensar que estaba mal generada pero después de verificar el proceso varias veces. Los resultados del algoritmo

LVQ muestran que aunque la calidad de la clasificación de la SOM no era elevada, es un gran sistema de clasificación para la posterior aplicación de algoritmos más complejos.

En total se obtuvieron 82 modelos, pero como no tiene mucho sentido mostrarlos todos solo se incluye una tabla con los mejores resultados:

Nº LVQ	alpha	Dec_alpha	win	nPas	Sensibilidad	Susceptibilidad	Eficacia
19	0.01	0.9	0.8	25	0.81	0.28	0.76
22	0.01	0.9	0.8	50	0.80	0.27	0.76
28	0.01	0.95	0.8	50	0.82	0.28	0.77

Tabla 8 - Mejores resultados para la clasificación 1-NN de LVQ

6.4 RESULTADOS MCE + 1-NN

El MCE era el último algoritmo que iba a probarse. Era el paso final en el proyecto y se buscaba una mejora sobre el resultado obtenido para LVQ.

MCE es un algoritmo ligeramente más complejo que LVQ y aunque su base de funcionamiento no es muy diferente, la característica de minimización de los errores de MCE debían proporcionar unas sutiles diferencias positivas.

Por decirlo de una forma más simple, MCE no va a introducir grandes mejoras sobre LVQ, simplemente va a ayudar a definir y perfeccionar los límites de cada neurona para que las clasificaciones tengan mayor eficacia.

Esto se puede observar en la tabla que se incluye a continuación. La sensibilidad y eficacia del sistema son muy similares a las obtenidas para LVQ con todos los valores en torno al 80% para sensibilidad y 77% para eficacia. Sin embargo la diferencia se observa en la susceptibilidad ya que la tasa de falsos positivos es de media menor que para el resto de modelos.

Nº	alpha	Dec alpha	pSi	Inc pSi	Pasadas	Sensibilidad	Susceptibilidad	Eficacia
66	0.01	0.99	0.96	1.3	75	0.80	0.23	0.77
71	0.01	0.99	0.99	1.3	50	0.81	0.23	0.76
72	0.01	0.99	0.99	1.3	75	0.81	0.24	0.76

Tabla 9 - Mejores resultados para la clasificación 1-NN de MCE

Estos resultados convierten a MCE en el mejor de los algoritmos probados para la generación del modelo de detección del abandono de clientes

CAPÍTULO 7. CONCLUSIONES Y TRABAJOS FUTUROS

Aunque los resultados obtenidos no han sido tan buenos como inicialmente se esperaba, considero que el resultado global del proyecto es positivo, ya que se ha conseguido generar diferentes modelos de predicción usando distintos algoritmos y se ha podido ver la evolución de los resultados conforme a lo esperado según la complejidad de los mismos.

Si no hubiera habido retrasos podría haberse ampliado la batería de pruebas y quizá se habría llegado a un modelo ligeramente mejor. Sin embargo, esto tampoco es una certeza ya que aunque una vez se detectó el fallo se redujo la batería de experimentos, se mantuvo la amplitud del rango de parámetros para comprobar la sensibilidad del modelo.

Aunque el sistema no es perfecto, podría llegar a implementarse, ya que como se ha dicho anteriormente, en este modelo de negocio en concreto, una sensibilidad de más del 80% con unos falsos positivos de en torno al 20% seguirían reportando amplios beneficios económicos a la empresa.

Respecto al beneficio que podría tener una aplicación real de este proyecto destacar que una empresa típica de este sector tiene una rotación anual del 20% de los clientes, es decir, el 20% de los clientes se pierden todos los años. Poniendo cifras a esto, si en el momento del análisis la empresa tenía unos 45.000 clientes activos, el 20% de esos clientes se irían durante ese año. Nuestro sistema tiene una sensibilidad aproximada del 80% por lo que de estos 9.000 podríamos detectar 720. Aunque sólo se consiguiera retener a la mitad de esos clientes, el impacto económico que tendría sobre la empresa sería muy destacable.

Como trabajos futuros podrían incluirse los siguientes:

- Pruebas con nuevos algoritmos como Historias Vecinas para ver si se mejora aún más el resultado obtenido por el algoritmo MCE.
- También podría repetirse la batería de experimentos pero de manera más exhaustiva para poder asegurarse que se obtiene el óptimo para los parámetros variados.

- Arrancar proyecto piloto con datos dinámicos de la empresa para ver los resultados y el impacto económico.

Una de las principales ampliaciones que podría llevarse a cabo es aplicar el modelo con datos de otro sector o de otra empresa, para ver si podría reciclarse el modelo y si los resultados se mantienen, y si no, cuanto se tardaría de verdad en adaptar completamente el modelo y en volver a realizar todo el proceso.

CAPÍTULO 8. BIBLIOGRAFÍA

[1] Ruiz, Sergio. "El algoritmo k-NN y su importancia en el modelado de datos". AD TECH & ANALYTICS. ANALYTICA WEB. Julio 2017. https://www.analiticaweb.es/algoritmo-knn-modelado-datos/.

[2] "Mapa auto-organizado". https://es.wikipedia.org/wiki/Mapa_autoorganizado.

[3] Ortiz, Juan & Peña Cuellar, David & Espitia, Helbert. "Análisis del efecto-día en el Mercado accionario colombiano empleando mapas auto organizados". 2014. https://www.researchgate.net/publication/279297177_Analisis_del_efecto-dia_en_el_mercado_accionario_colombiano_empleando_mapas_autoorganizados?_sg=EbfJYrkSEEQ0-WJXr4n4An2LxTjDfPeibsg8Q4M3ZtP_7jpGg6nuKxcyBavq38T5lYWyR2kkvF2URaARlS_fqVJF4Gkz54oCDQ.

[4] Brownlee, Jason. "LEARNING VECTOR QUANTIZATION FOR MACHINE LEARNING". Machine Learning Algorithms. Abril, 2013. https://machinelearningmastery.com/learning-vector-quantization-for-machine-learning/ .

[5] Crone, Sven & Lessmann, Stefan & Stahlbock, Robert. "Support vector machines versus Artificial Neural Networs – new potential in data mining for customer relationship management?". 2018. https://www.researchgate.net/publication/267252990_SUPPORT_VECTOR_MACHINES_VERSUS_ARTIFICIAL_NEURAL_NETWORKS_-_NEW_POTENTIAL_IN_DATA_MINING_FOR_CUSTOMER_RELATIONSHIP_MANAGEMENT?_sg=e870FweibnyaUFnm2o-9IWoPVT-vAAGgajpSJIO43qvwKTD-LjVhxYgCvi6-vP4XOHp5Ks8_1M.

[6] Liu, Cheng-Lin & Nakagawa, Masaki. "Evaluation of prototype Learning algorithms for nearest-neighbor classifier in application to handwritten character recognition". Enero 2000.

[7] Maynez, Natalia, "19 Tecnologías de Inteligencia Artificial que dominarán el 2018". Blog Adext. Septiembre 2017. https://blog.adext.com/es/tecnologias-inteligencia-artificial-2018.

[8] "Theoretical explanation of how siri Works". NdimensionZ. Abril 2017. http://www.ndimensionz.com/kb/theoretical-explanation-of-how-siri-works/.

[9] "Fingerprint Technology Overview". IdentityOne. http://www.identityone.net/BiometricTechnology.aspx.

[10] "7 empresas que están haciendo maravillas con la IA". INSIDER PRO. Abril 2016. https://es.insider.pro/investment/2016-04-19/7-empresas-que-estan-haciendo-maravillas-con-la-ia/ .

[11] Thompson, Cadie. "How Google's self-driving cars see the World". Business Insider UK, Octubre 2015. http://uk.businessinsider.com/how-googles-self-driving-cars-see-the-world-2015-10?IR=T.

[12] "The 2016 AI Recap: Startups See Record High In Deals And Funding". Research Briefs CBInsights. Enero 2017. https://www.cbinsights.com/research/artificial-intelligence-startup-funding/.

[13] Pavón, Fernando. "Generación de conocimiento basado en Aprendizaje Automático y Aplicación en Diferentes Sectores, AIPAKA". Tesis Doctoral. Febrero 2016. https://www.gamco.es/media/uploads/2016/05/16/Resumen_Tesis_Obtencion-automatica-de-conocimiento-a-partir-de-historicos.pdf.

[14] Juliá, Samuel, "Ventajas e inconvenientes de usar Linux en tu empresa". Gadae Blog. http://www.gadae.com/blog/ventajas-utilizar-linux/ .

[15] Logo GitLab. https://www.pinterest.es/pin/796785359046486276/?lp=true.

[16] Logo Evernote https://treasurewebdesigns.com/evernote/.