Trabajo Fin de Grado

Técnicas de aprendizaje automático para predicción de dianas de microRNA

Presentado por:
Violeta Bastida Alba

Tutor:
Dr. Rafael Blanquero Bravo

UNIVERSIDAD D SEVILLA

20 de junio de 2016

Índice de contenidos

Capítulo 1

Introducción

Los microRNA (miRNA) son pequeñas moléculas de RNA que no intervienen directamente en la formación de proteínas, ya que actúan como moléculas reguladoras de la transcripción génica mediante la degradación del RNA mensajero o la inhibición de la traducción. En recientes estudios bioinformáticos se ha descubierto que un miRNA puede controlar un gran número de genes diana y, por lo tanto, la caracterización y localización de estos genes diana puede ser crucial para identificar el papel que los miRNA pueden ejercer como genes supresores de tumores.

Diversos estudios han demostrado que la expresión de los miRNA se encuentra alterada en enfermedades como el cáncer y su perfil de expresión nos puede servir para diagnosticar y pronosticar distintos tipos de neoplasias. Los estudios sobre las funciones reguladoras de los miRNA han demostrado que tienen un papel crítico en procesos biológicos como son el control de la proliferación celular, la apoptosis, la angiogénesis, o el proceso de metástasis. En el futuro, es posible que la manipulación de la regulación en la expresión de los miRNA, mediante inhibición o sobreexpresión sintética, nos permita desarrollar nuevos tratamientos para estas enfermedades [4, 49].

Un paso necesario en la consecución de este objetivo consiste en la identificación de los RNA mensajeros diana de cada miRNA. Para conseguir esta identificación resultan especialmente útiles numerosas técnicas de aprendizaje automático, principalmente clasificadores, cuya revisión se realiza en el siguiente trabajo fin de grado.

Capítulo 2

La información genética y su expresión

2.1. La información genética

La información genética necesaria para la formación y el funcionamiento de un ser vivo reside en las moléculas de DNA (ácido desoxirribonucléico) en forma de genes, de manera que un gen es un fragmento de molécula de DNA.

El DNA y el RNA (ácido ribonucléico) son los dos tipos de ácidos nucleicos existentes en los organismos y, junto con hidratos de carbono, lípidos y proteínas, forman las cuatro clases de biomoléculas que caracterizan la vida. Los genes pueden entenderse como las unidades básicas de almacenamiento de la información genética, pudiendo ser expresados, silenciados, sufrir mutaciones y replicaciones. Cada gen contiene información para una determinada tarea concreta como puede ser el desarrollo de una función fisiológica en un ser vivo mediante la síntesis de una proteína. El conjunto de todos los genes de un ser vivo se denomina genoma y es trasmitido de organismos adultos a su descendencia durante el proceso de reproducción. El lugar en el que reside la información genética de un ser vivo depende del tipo de célula. En los organismos eucariotas, cuyas células tienen núcleo diferenciado, el material genético se encuentra en el núcleo celular empaquetado en estructuras discretas denominadas cromosomas. Por el contrario, en los organismos procariotas, con células carentes de núcleo, el DNA forma una molécula única denominada nucleoide. En los organismos eucariotas, cada cromosoma consiste en una molécula de DNA bicatenario asociada con proteínas básicas denominadas histonas, y con otras proteínas no histónicas. La función de las histonas es la de constituir el soporte estructural del DNA en una fibra de estructura compleja, la

cromatina.

Todas las células de un organismo contienen la misma información genética. Esta puede aparecer de dos formas distintas según el tipo de células que se considere, haploide o diploide. En las células haploides, como por ejemplo las células reproductoras, solo aparece una copia de la información genética mientras que las células diploides , que forman los tejidos de los organismos complejos como animales y plantas, contienen dos copias, debido a la presencia de pares de cromosomas homólogos.

2.2. Almacenamiento de la información genética

Los ácidos nucleicos (DNA y RNA) tienen naturaleza polimérica, estando formados por bloques químicos llamados nucleótidos. Cada nucleótido presenta tres componentes diferenciados:

- Un grupo fosfato , PO_4^{3-}. Cada nucleótido puede contener uno, dos o tres grupos fosfato.
- Una pentosa, es decir, un monosacárido de cinco átomos de carbono. En los nucleótidos de RNA la pentosa es la ribosa, mientras que en los de DNA es la desoxirribosa.
- Una base nitrogenada que pueden ser: Adenina(A), Citosina(C) y Guanina(G), presentes tanto en DNA como en RNA, Timina(T), únicamente presente en DNA, y Uracilo (U), sólo presente en RNA.

La unión entre nucleótidos se lleva a cabo mediante enlaces fosfodiéster, en los que un grupo fosfato se una al carbono $5'$ de la pentosa de un nucleótido y, simultáneamente, al carbono $3'$ de la pentosa del siguiente nucleótido (Figura 2.2). La adición de nucleótidos está regulada por la enzima DNA Polimerasa. Esta enzima es esencial en la fase de replicación de DNA.
El emparejamiento entre dos cadenas de nucléotidos para formar la doble cadena de DNA se realiza a través de puentes de hidrógeno entre las bases nitrogenadas, emparejándose éstas de la siguiente forma: A-T, G-C en DNA, asociándose las dos cadenas en sentidos opuestos (Figura 2.1) , y A-U, G-C en RNA. El orden, o la secuencia, en la que se encuentran estas bases determina qué instrucciones o información biológica está recogida en esa cadena de DNA. Por ejemplo, la secuencia ATCGTT puede que lleve las instrucciones para poseer ojos azules, mientras que la secuencia ATCGCT puede desembocar en ojos marrones.

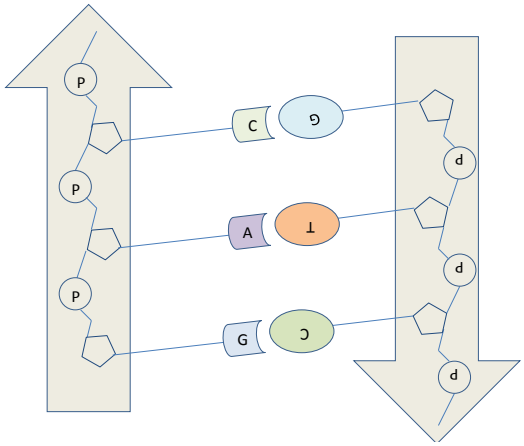

Figura 2.1: En una molécula de DNA, los esqueletos de fosfatos de azúcar de las dos cadenas están orientados en sentidos diferentes.

Además, la doble cadena de DNA adquiere una forma de doble hélice, que podría entenderse como una escalera donde los pares de bases forman los peldaños y el monosacárido y el grupo fosfato, las paredes laterales.

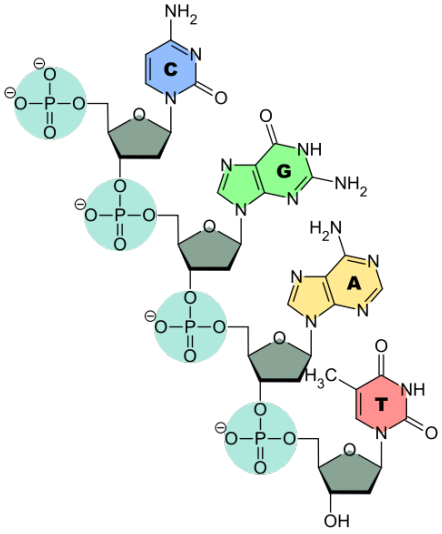

Figura 2.2: Representación de una cadena de nucleótidos unidos mediante enlaces fosfodiéster. Se puede apreciar el grupo fosfato, la pentosa y la base nitrogenada de cada nucleótido constituyente.

2.3. Expresión de la información genética: Transcripción y traducción

Las proteínas son las moléculas complejas encargadas de llevar a cabo las funciones de control del desarrollo y funcionamiento en un organismo. Así pues, las instrucciones que residen en el DNA deben ser convertidas en mensajes que deben ser interpretados para llevar a cabo la síntesis de proteínas. En este proceso de interpretación de la información genética es donde interviene el RNA.

El RNA puede entenderse como la molécula intermediaria que porta la información genética que reside en el DNA presente en el núcleo celular hasta los lugares de síntesis de proteínas, los ribosomas en el citoplasma. Existen diferentes tipos de RNA según la función que llevan a cabo. Pueden distinguirse dos tipos de RNA: RNA codificantes, que son los encargados de transmitir la información para la síntesis de proteínas, y RNA no codificantes. Este último tipo está formado por RNA cuya misión es la regulación de la expresión génica.

La transmisión de la información genética se divide en dos etapas fundamentales: Transcripción y Traducción. Este flujo de información unidireccional se conoce como el *dogma central de la genética*. (Figura 2.3)

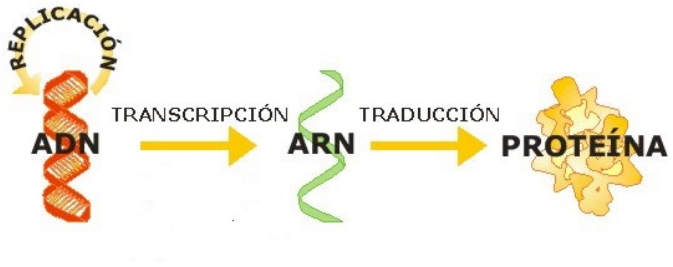

Figura 2.3: Transcripción y traducción. Fases fundamentales en la expresión de la información genética.

2.3.1. Transcripción

La transcripción de un gen comienza cuando una enzima, la RNA-polimerasa, se une al DNA en un sitio específico denonimado promotor, una secuencia de varias docenas de pares de nucleótidos localizada en un extremo de un gen. Entonces, las dos hebras

de DNA se desenrollan parcialmente, y la RNA-polimerasa comienza a desplazarse a lo largo de la sección de DNA donde se encuentra el gen. A medida que la RNA-polimerasa se mueve, sintentiza un tramo de RNA utilizando una de las hebras de DNA como molde. Los nucleótidos que se añaden al RNA son complementarios a los nucleótidos de la hebra de DNA molde.

De este modo, la secuencia de nucleótidos específica que fabrica un gen se transcribe a una secuencia de nucleótidos complementaria en el RNA.

La transcripción concluye cuando la RNA-polimerasa llega a un sitio llamado terminador, cerca del extremo opuesto del gen. En este punto, la RNA-polimerasa libera el nuevo RNA fabricado y se separa del DNA, volviéndose a enrollar las dos hebras de DNA.

Antes de iniciarse el proceso de traducción, el RNA fabricado ha de ser procesado y retirado del núcleo.

Primeramente, una molécula de guanosín trifosfato (GTP) se añade aun extremo de RNA. Este GTP añadido se denomina caperuza o cap. En segundo lugar, se añaden entre 50 y 200 nucleótidos al extremo opuesto del RNA. Esta secuencia de nucleótidos se conoce como cola poli-A (poli adenina). La cola poli-A podría promover el transporte del RNA desde el núcleo hasta el citoplasma. Posteriormente a la entrada del RNA en el citoplasma, la caperuza y la cola poli-A ayudan a unir el RNA al ribosoma y protegen al RNA de la digestión enzimática.

Sólo una vez que el RNA ha sido procesado por completo y ha accedido al interior del citoplasma, se dice que es RNA mensajero (RNAm). Entonces está listo para ser traducido, [42]. (Figura 2.4)

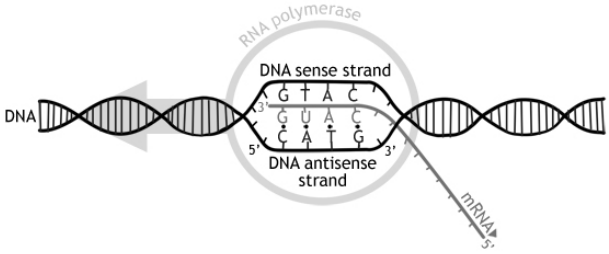

Figura 2.4: Síntesis de RNAm a partir de una cadena de DNA molde.

2.3.2. Traducción

La traducción se lleva a cabo en el citoplasma, interviniendo lo que se conoce como *código genético*. Podemos entender este *código genético* como un diccionario que establece la relación entre la información que porta el RNAm y el lenguaje de la fabricación de proteínas.

Durante este proceso, la secuencia de nucleótidos de RNAm se convierte en una secuencia de aminoácidos de una proteína de acuerdo con las reglas especificadas por el *código genético*. Durante esta fase, los nucleótidos de RNA son interpretados por la maquinaria traductora de los ribosomas en paquetes de tres nucleótidos. Cada uno de estos paquetes, llamados codones, codifica para un aminoácido específico.

La traducción precisa de la participación de los ribosomas, que están compuestos por proteínas y por otro tipo de RNA denominado RNA ribosómico (RNAr). Para este proceso, también son necesarios el RNAm y un tercer tipo de RNA, el RNA de transferencia o transferente (RNAt), [42].

2.4. La regulación de la expresión genética: miRNA

Aunque cada tipo de célula contiene todos los genes del organismo, no los expresa todos. Por el contrario, las células expresan sólo aquellos genes que codifican para las proteínas que necesitan. Esta selección de ciertas partes de la información genética de un organismo se denomina *expresión genética diferencial*.

Las células ejercen la expresión genética diferencial en cada paso comprendido entre la lectura del código genético y la utilización de proteínas. Se impide o desactiva la transcripción para algunos genes, y se activa o pone en marcha para otros. El procesamiento del nuevo RNA favricado para su conversión en RNAm puede ser facilitado o inhibido. Una vez que el RNAm se introduce en el citoplasma, las enzimas pueden traducirlo, ignorarlo o destruirlo, [42].

Los estudios mas recientes sobre la actividad transcripcional del genoma humano revelan que aproximadamente solo se transcribe a RNA la mitad del DNA y un gran porcentaje de este RNA transcrito representa RNA no codificante pero que igualmente juega un papel activo muy importante, [46]. En los organismos eucariotas existe un repertorio muy variado de RNA no codificante de tamaño pequeño de aproximadamente 21-25 nucleótidos cuyo papel fundamental es la regulación de la expresión génica; por lo tanto, su presencia es determinante en procesos de diferenciación celular y desarrollo, proliferación, adquisición y mantenimiento de un fenotipo dado, [37].

Estos RNA pequeños se asocian a complejos enzimáticos y son guiados para el acoplamiento a secuencias complementarias de RNAm, conocidas de determinados RNAm diana. Esta interacción funcional entre ambos puede derivar en muchos casos en la degradación del RNAm y, por tanto, en la represión traduccional, [56].

Los microRNA (miRNA) representan un grupo de estos RNA no codificantes de secuencia corta (18-25 nucleótidos) encargados de importantes funciones reguladoras.

El primer miRNA descrito fue *line-4*, que regula los diferentes pasos en el desarrollo del gusano *Caenorhabditis elegans*, [34]. Desde entonces gracias al esfuerzo de la biología molecular y a las herramientas de predicción bioinformática, se han descubierto un gran número de miRNAs en diversas especies, incluyendo mamíferos y específicamente, humanos, [3].

El descubrimiento de los mecanismos de regulación de la expresión genética ha permitido que se desarrollen numerosas alternativas para el estudio, diagnóstico y tratamiento de diversas enfermedades reguladas por alteraciones en la expresión genética. La importancia de estos descubrimientos radica en determinar la "firma genética" basada en miRNA, [37].

El *miR-175*, que se encuentra expresado en altos niveles en el páncreas, es requerido para la homeostasis de la glucosa. Los animales que carecen de este miRNA presentan un fenotipo hiperglicémico; mientras que en los animales obesos carentes del mismo miRNA disminuye considerablemente la capacidad endocrina del páncreas.

De igual manera, la sobreexpresión del miRNA *miR-30d* incrementa la expresión de insulina, mientras que su inhibición bloquea la transcripción del gen de insulina estimulada por la glucosa. Estos y otros muchos hallazgos son muy importantes para el estudio de enfermedades como la diabetes mellitus y de sus correspondientes tratamientos terapéuticos, [45].

Una propuesta terapéutica reciente es el empleo del *miR-203* en procedimientos de terapia genética para el tratamiento de enfermedades oncogenéticas en las que la expresión del gen ABL, implicado en procesos de diferenciación celular y por lo tanto en procesos de metástasis, juegan un papel fundamental. Tras un extenso estudio se concluye que el gen ABL es el gen diana del miRNA *miR203*, [10].

2.4.1. Biosíntesis de los microRNA

La biosíntesis de los miRNA incluye varias etapas, [5]. Inicialmente, los miRNA son transcritos por la enzima RNA polimerasa II para generar moléculas precursoras o pri-miRNA. Estos pri-miRNAs se autocomplementan en estructuras en forma de horquilla

compuesta de un tallo y un bucle.

Posteriormente, estos pri-miRNA son procesados en el núcleo por el complejo proteico conocido como microprocesador para generar un precursor aun más pequeño, pre-miRNA, de aproximadamente 65 nuncleótidos (nt) de longitud aproximadamente. Este microprocesador está formado por la enzima Drosha y la proteína de unión a RNA de doble cadena, denominada DGCR8 en humanos. En este paso de maduración de los pri-miRNAs, Drosha fragmenta el tallo de la estructura y permite su liberación. La proteína DGCR8 juega un papel importante en el reconocimiento de los sitios de corte de Drosha en el tallo de la estructura del pri-miRNA.

Después de esta etapa de procesamiento dentro el núcleo, los pre-miRNA son reconocidos por el factor nuclear de exportación Exportina-5 (Exp5), que junto con otras proteínas de unión, forman un complejo de transporte nuclear que conduce a los pre-miRNA a través de los poros del núcleo hacia el citoplasma. Una vez ubicados en el citoplasma, estos pre-miRNAs son reconocidos por un complejo de procesamiento compuesto por una enzima llamada Dicer. Esta enzima se encarga de cortar el tallo y bucle del pre-miRNA para generar un RNA dúplex formado por una cadena de miRNA maduro y una cadena de miRNA complementaria de aproximadamente 22 nt de longitud.

Acto seguido, el miRNA maduro es incorporado al complejo silenciador RISC (RNA Induced Silenced Complex) (Figura 2.5).

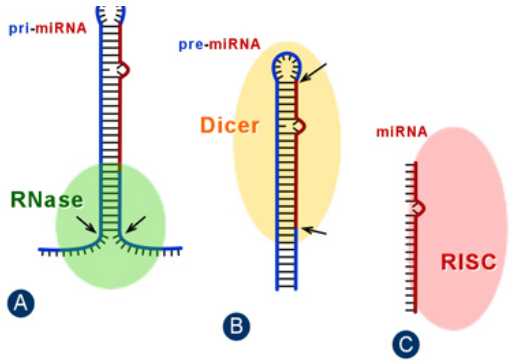

Figura 2.5: Biosíntesis de miRNA.

Este complejo RISC está compuesto por varias proteínas, entre ellas Dicer, TRBP (transactivation-response element RNA-binding protein) y, principalmente, por las proteínas Argonauta.

Las proteínas Argonauta están localizadas en regiones específicas del citoplasma

denominados cuerpos-P, que pueden ser definidos como sitios de almacenamiento de RNAm. Este complejo RISC se ensambla con la proteína Argonauta, para formar el complejo silenciador inducido por microRNAs (miRISC, del inglés miRNA-induced silencing complex), el cual selecciona la cadena madura o guía. Esta cadena guía es la responsable de dirigir el proceso de silenciamiento. La cadena que no se incorpora al complejo RISC es degradada. (Figura 2.6)

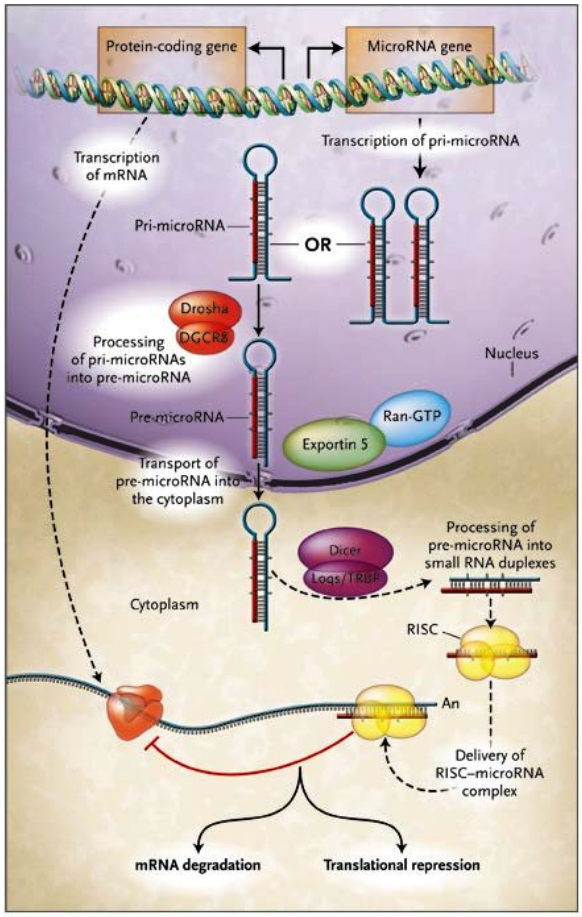

Figura 2.6: Proceso completo de biosítesis de miRNA.

2.4.2. Acoplamiento miRNA-RNAm

El acoplamiento entre una hebra de miRNA y su diana de RNAm se lleva a cabo a través de la región 3' UTR del RNAm. En genética se denomina UTR (del inglés, untranslated region) a las regiones no traducidas de los genes. Se habla generalmente de un 5' UTR y de un 3' UTR. Las UTRs poseen gran importancia en la regulación de la expresión génica. Existen proteínas adaptadoras que reconocen secuencias específicas, no codificantes, del 3' UTR y además, están implicadas en la correcta expresión espacial y temporal de los genes. Las secuencias responsables del acortamiento de la vida media de los RNAm están en las regiones 3' UTR. La regulación se hace a través de acoplamiento de factores que se unen a esas secuencias características. (Figura 2.7) El

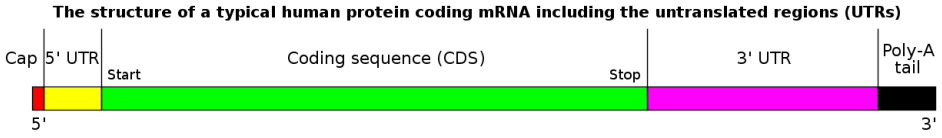

Figura 2.7: Estructura típica de una molécula de RNAm. Se puede apreciar la caperuza de GDP en el extremo $5'$ y la cola poli-A en el extremo $3'$.

emparejamiento entre miRNA y su diana se lleva a cabo de manera idéntica al emparejamiento de moléculas de DNA y RNA, esto es, se unen mediante complementariedad de las bases nitrogenadas, [9]. Al contrario de lo que ocurre con el emparejamiento entre DNA y RNA, este proceso puede o no ser canónico, esto es, pueden existir bases que queden desemparejadas (Figura 2.8).

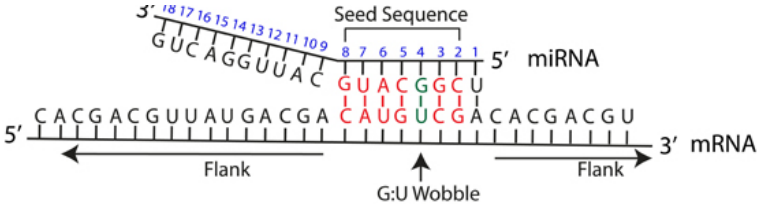

Figura 2.8: Emparejamiento miRNA y su diana.

2.4.3. Estructura de un acoplamiento miRNA-RNAm diana.

Una vez formado el dúplex entre la hebra madura de miRNA y su diana de RNAm, hemos de considerar qué forma o qué estructura posee esta formación.

La zona más importante, y que además juega un papel determinante en la selección del sitio donde va a producirse el acoplamiento entre el miRNA y su diana, se conoce como *región Seed*. Esta región incluye de 6 a 8 nt en el extremo 5' de la hebra de miRNA y se caracteriza por ser la zona donde el acoplamiento se produce de forma complementaria o canónica. (Figura 2.8).

Por otra parte, la *región outSeed* es aquella secuencia del emparejamiento donde nos encontramos bases desemparejadas que dan lugar a bucles o loops. (Figura 2.8)

2.4.4. Mecanismo de funcionamiento de los microRNA

El mecanismo de funcionamiento de los miRNA depende, en gran medida, del grado de complementariedad con su RNAm diana. Si la complementariedad entre ambas secuencias es completa, el RNAm diana será degradado por el complejo RISC. Sin embargo, si el emparejamiento entre las bases no es perfecto, como ocurre con la mayoría de los miRNA de mamíferos, se producirá la inhibición de la traducción (silenciamiento).

Un miRNA puede tener muchos RNAm diana y cada RNAm puede estar regulado por varios miRNA. Entre los mecanismos que se han propuesto que pueden resultar de esta interacción entre miRNA y su diana se encuentran: bloqueo del inicio de la traducción, represión pos-inicio de la traducción, y desestabilización del RNAm diana a través de un proceso de deadenilación. Estos procesos, aparentemente distintos entre sí, no son mutuamente excluyentes. En estos tres procesos, los RNAm diana son secuestrados en los cuerpos-P para su almacenamiento o degradación. Sin embargo, bajo condiciones de estrés estos RNAm almacenados pueden ser liberados y, finalmente, traducidos, [6, 42, 3, 11, 15, 38].

En resumen, los miRNA participan activamente en la regulación de la expresión genética actuando sobre los RNAm silenciando, al menos temporalmente, la traducción de la información genética que porta. Un cambio en las condiciones ambientales pueden inducir a la traducción. Los miRNA, por otro lado, parecen tener implicaciones directas en el control de enfermedades de carácter genético, como el cáncer y diabetes, entre otras. La falta de emparejamiento canónico entre miRNA y RNAm supone un reto de cara a identificar los RNAm diana de cada miRNA.

Capítulo 3

Aprendizaje automático. El problema de clasificación

Desde un punto de vista básico, el principal objetivo del *Aprendizaje* consiste en analizar datos para extraer conocimiento. Este conocimiento puede ser en forma de relaciones, reglas o patrones inferidos de datos previamente conocidos. Por ejemplo, dada una muestra de tamaño finito de individuos de una determinada población, se pretende encontrar una clasifiación de los mismos en diferentes clases o grupos según sus características comunes o no. El objetivo del **Aprendizaje automático**, o *Machine Learning* en inglés, es, por tanto, desarrollar algoritmos para los ordenadores de manera que éstos sean capaces de generalizar comportamientos y reconocer patrones comunes a partir de información suministrada previamente.

Supongamos que se dispone de una población en la que se conocen las características de sus individuos, es decir, las clases a las que pertenecen. Este problema se conoce como **aprendizaje supervisado** ya que los elementos que se tratan ya están asociados con un valor objetivo (una clase determinada, un valor real, etc.) que conocemos previamente. Como ejemplo destacado de aprendizaje supervisado podemos considerar el problema de clasificación, que se analizará más adelante. De igual manera, cuando no se dispone previamente de una batería de elementos asociados con un valor objetivo, estamos considerando problemas de **aprendizaje no supervisado**.

Desde el punto de vista estadístico, el problema del aprendizaje automático puede interpretarse como un problema general de inferencia estadística. Esto es, se parte de observaciones particulares y se llega a descripciones generales con la construcción de un modelo. La principal diferencia entre el enfoque estadístico y el del aprendizaje automático reside en que la inferencia estadística considera una descripción de los datos en términos de medidas de probabilidad en lugar de un enfoque determinista

como puede ser una función de predicción, en el caso del aprendizaje automático. Aun así, los problemas a resolver en ambos casos son prácticamente equivalentes, [54].

Existen muchas formas diferentes de representar los modelos y cada una de ellas determina el tipo de técnica que puede usarse para inferirlos.

En la práctica, los modelos pueden ser de dos tipos: *predictivos y descriptivos*. Los modelos predictivos pretenden estimar valores futuros o desconocidos de variables de interés, como es el problema de clasificación y el de regresión. Los modelos descriptivos, sin embargo, identifican patrones que explican o resumen los datos, es decir, sirven para explorar las propiedades de los datos examinados pero no para predecir nuevos datos. Dentro de éstos podemos destacar los problemas de agrupamiento y las reglas de asociación.

Dentro del Aprendizaje Automático se consideran diferentes tipos de problemas, donde cada uno de ellos es resuelto por un algoritmo. Algunos problemas usuales, ya mencionados previamente, son:

- **Problema de clasificación**: El objetivo de este problema es intentar predecir, sobre un conjunto de clases prefijadas, aquellas a las que pertenecen nuevos objetos.

  Ejemplo. Supongamos que un médico pretende clasificar a nuevos pacientes en dos clases, "enfermo" y "no enfermo". Para ello dispone de una base de datos de antiguos pacientes clasificados en estas dos clases según los parámetros hallados en sus respectivas analíticas de sangre.

- **Problema de regresión**: Es un problema cuyo principal objetivo es predecir un valor real. Esta es la principal diferencia respecto al problema de clasificación, pues el valor a predecir es una magnitud numérica continua, mientras que en el problema de clasificación es una magnitud discreta. El objetivo que se persigue en el problema de regresión es pronosticar ese valor de modo que se minimice el error o diferencia entre el valor predicho y el real.

  Ejemplo. Se pretende fijar el precio de una casa basándose en los metros cuadrados que tiene, número de habitaciones, cuartos de baños, etc. Se busca un algoritmo que devuelva el precio de la casa en euros, un valor real.

- **Problema de agrupamiento o *clustering***: Este problema consiste en obtener grupos a partir de los datos. La diferencia entre el problema de clasificación y el de agrupamiento radica en que este último analiza los datos para posteriormente crear unos determinados grupos donde incluirlos. mientras que en el problema de clasificación los grupos o clases están predefinidos. Los datos son agrupados tratando de hacer máxima la similaridad entre los elementos de un mismo grupo

y, al mismo tiempo, minimizando la similaridad entre los distintos grupos. En otras palabras, se trata de crear grupos homogéneos con máxima heterogeneidad entre ellos.

Ejemplo. Una tienda que ofrece sus servicios a través de internet usa técnicas de clustering para identificar grupos de clientes en base a sus preferencias de compras, permitiéndole así ofrecer un servicio personalizado.

3.1. El problema de clasificación

Como hemos indicado anteriormente, el problema de clasificación se enmarca dentro del aprendizaje supervisado. Supongamos que tenemos una población dividida en m clases mutuamente excluyentes y previamente conocidas,

$$C = \{c_1, c_2, c_3, ..., c_m\}.$$

Cada elemento de la población X viene caracterizado por un vector n-dimensional, conocido como **vector de características**:

$$x_i = (x_{i1}, x_{i2}, x_{i3}, ..., x_{in})$$

Supondremos que se dispone de lo que se conoce como **muestra de aprendizaje**, que es un subconjunto de la población para el cual se dispone del vector de características de cada uno de sus elementos, así como de la clase a la que pertenecen. Cada elemento de la muestra de aprendizaje se denomina **ejemplo** o **instancia**.

$$T = \{(x_1, c_1), (x_2, c_2), (x_3, c_3), ..., (x_p, c_p)\} \qquad x_i \in X, c_i \in C$$

El objetivo del problema de clasificación consiste en diseñar procedimientos que permitan decidir, a partir de la muestra de aprendizaje, a qué clase $c \in C$ debe asignarse un nuevo elemento $x \in X$. Se trataría, por tanto, de encontrar una función de asignación de clases:

$$\begin{aligned} g : X &\to C \\ x &\mapsto c \end{aligned}$$

De especial importancia es el problema de clasificación binario, donde C consta sólo de dos elementos, uno de ellos interpretado como la clase positiva y el otro como la clase negativa. Por ejemplo, clasificar en clases tipo “enfermo” frente a “no enfermo”, “válido” frente a “no válido”, “nocivo” frente a “no nocivo”, etc.

Existen muchos clasificadores o modelos de clasificación: k-NN, SVM, Naïve Bayes, análisis discriminante, redes neuronales, etc, todos ellos basados en diferentes conceptos, que pueden ser probabilísticos, en términos de distancia y proximidad, etc. La cuestión que surge una vez planteado el problema genérico de clasificación es el diseño de un clasificador adecuado para una situación concreta que se plantee.

3.2. Construcción de un clasificador. Validación

El problema que surge a la hora de obtener un modelo de clasificación radica en que, generalmente, el modelo obtenido clasifica muy bien los elementos de la muestra de aprendizaje, pero falla al clasificar nuevos objetos, de manera que, en realidad, no se tiene una medida objetiva de la efectividad del modelo. Este problema se conoce como **sobreajuste o sobreaprendizaje**.

Para evitar este problema se utilizará un conjunto de datos para evaluar el rendimiento del clasificador distinto de aquel que se ha utilizado para construirlo. Surge así el concepto de **muestra de validación**, constituída por instancias para las que se conoce tanto su vector de características como su clase, al igual que en el caso de la muestra de aprendizaje (puede ser un subconjunto de ésta). Este nuevo conjunto se utilizará para evaluar la efectividad o rendimiento del modelo; al no haber intervenido la muestra de validación en el proceso de construcción del clasificador, es de esperar que proporcione resultados de efectividad no sesgados.

A la hora de validar un clasificador, es decir, de medir su efectividad, podemos considerar varias técnicas:

- **Validación Simple**: Conocida también como *Método H (hold out)* o *Training-test*. Este método de validación consiste en una división aleatoria de la base de datos que se posee en dos grupos; un grupo se toma como muestra de aprendizaje, y se utiliza pues para construir el modelo, y el restante como muestra de validación, usado para la evaluación del modelo. La división usual es 2/3 para muestra de aprendizaje y 1/3 para muestra de validación (Figura 3.1).

  El inconveniente que presenta esta técnica de validación es que la base de datos debe tener un tamaño considerable para que tenga sentido esta división, ya que supone prescindir de una proporción importante de observaciones a la hora de construir el clasificador.

- **Validación cruzada con k pliegues**: En inglés, *k-fold cross validation*. Para esta técnica, se divide aleatoriamente la base de datos en k grupos del mismo

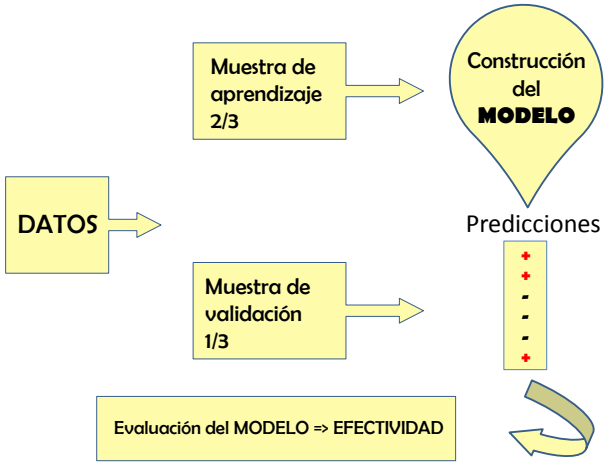

Figura 3.1: *Training-test.*

tamaño. Para cada uno de estos grupos i, $i = 1, \ldots, k$, los $k-1$ restantes se usan para construcción del modelo y el grupo i para validación (Figura 3.2).

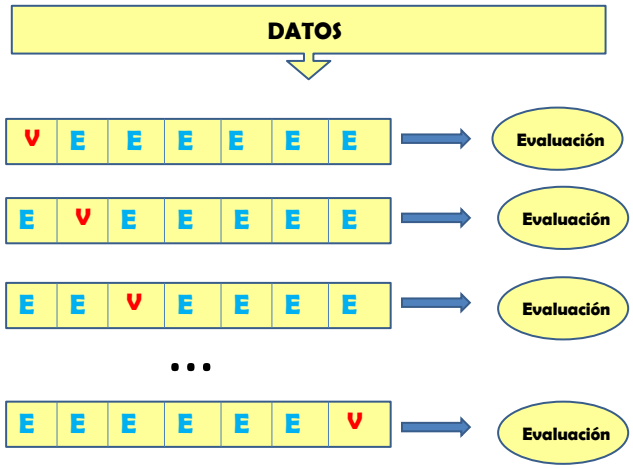

Figura 3.2: *k-fold cross validation.*

- ***Leave-one-out***: Esta técnica es un caso particular de validación cruzada que consiste en considerar un número de pliegues igual al número de individuos de la base de datos. Por lo tanto, cada muestra de validación tiene tamaño uno y la construcción del modelo se lleva a cabo tantas veces como individuos posee la base de datos. Se trata de la técnica de validación que mejor uso hace de los

datos, pero conlleva un esfuerzo computacional muy alto. Esta técnica se usa con bases de datos de tamaño pequeño.

3.3. Validación del clasificador

Una vez construido el clasificador se debe valorar si proporciona resultados adecuados, es decir, si predice correctamente la clase de los elementos de la muestra de validación. Esta valoración se realiza empleando ciertas medidas de efectividad que tienen como base la **matriz de confusión**. La matriz de confusión permite la visualización del funcionamiento del clasificador, mostrando para ello de forma resumida la concordancia entre clase real y clase pronosticada. Cada **columna** representa el número de predicciones de cada clase y cada **fila** representa a los individuos o elementos en la clase real. La Figura 3.3 presenta un ejemplo de matriz de confusión.

	Naranjas	Manzanas	Plátanos	
Naranjas	5	3	0	Naranjas reales: 5+3+0=8
Manzanas	2	3	1	Manzanas reales: 2+3+1=6
Plátanos	0	2	11	Plátanos reales: 0+2+11=13

Figura 3.3: *Matriz de confusión.*

En la matriz del ejemplo, de las 8 naranjas reales que se tienen, el algoritmo predijo que 5 eran naranjas y 3 eran manzanas. De 6 manzanas que había, se clasificaron 2 como naranjas, 3 como manzanas y 1 como plátano. De igual manera, de 13 plátanos que había, el sistema clasificó 2 como manzanas y 11 como plátanos.

De aquí puede deducirse que el clasificador tiene problemas a la hora de distinguir entre naranjas y manzanas, pero distingue claramente mejor los plátanos del resto de frutas.

Como puede apreciarse en el ejemplo anterior, la matriz de confusión permite una aproximación visual de la precisión en las predicciones del clasificador.

Consideremos el caso de un problema de clasificación binario, en el que se tendrían dos clases, una **clase positiva (+)** y otra **clase negativa (-)**. En este caso, la matriz de confusión sería una matriz $A \in \mathcal{M}_{2\times 2}$. Del ejemplo anterior, podríamos extraer una matriz de confusión de un problema binario (Figura 3.4).

	Naranjas Clase positiva (+)	Manzanas Clase negativa (-)
Naranjas Clase positiva (+)	5	3
Manzanas Clase negativa (-)	2	3

Figura 3.4: *Matriz de confusión* de un problema binario.

Consideremos una matriz de confusión genérica para un problema de clasificación binario:

		Predicción	
		Clase positiva (+)	Clase negativa (-)
Realidad	Clase positiva (+)	TP	FN
	Clase negativa (-)	FP	TN

Tomando como punto de partida la matriz de confusión anterior, se definen diversos conceptos relacionados con la efectividad del clasificador:

- **Verdadero positivo (TP) y verdadero negativo (TN)**: los elementos $a_{1,1}$ y $a_{2,2}$ de la matriz de confusión corresponden al número de individuos o elementos que se han clasificado correctamente, es decir, la predicción del clasificador coincide con la realidad. Cuando un elemento es clasificado en la clase (+) y además en la realidad pertenece a la clase (+) tenemos un **verdadero positivo** y de igual manera en el caso de clasificar en la clase (-) tenemos un **verdadero negativo.**

- **Falso positivo (FP) y falso negativo (FN)**: Corresponden a los valores en las posiciones $a_{2,1}$ y $a_{1,2}$. Estos valores representan aquellos individuos que se han clasificado en una clase contraria a la que pertenecen en la realidad. Aquellos que se han clasificado en la clase (+) perteneciendo a la clase (-) representan un **falso positivo** y, de igual manera, aquellos elementos clasificados en la clase (-) cuando pertenecen a la clase (+) son **falsos negativos.**

- **Tasas**: A partir de la matriz de confusión se definen las siguientes tasas:

  - Tasa de verdaderos positivos (*True Positive Rate*)

$$TPR = \frac{TP}{TP + FN}$$

  - Tasa de verdaderos negativos (*True Negative Rate*)

$$TNR = \frac{TN}{TN + FP}$$

  - Tasa de falsos positivos (*False Positive Rate*)

$$FPR = \frac{FP}{FP + TN}$$

  - Tasa de falsos negativos (*False Negative Rate*)

$$FNR = \frac{FN}{TP + FN}$$

- **Sensibilidad y Especificidad**: En términos probabilísticos, la probabilidad de obtener un **verdadero positivo** se conoce como **sensiblidad**, mientras que la probabilidad de obtener un **verdadero negativo** se denomina **especificidad**. Sus estimadores vendrían dados por:

$$\text{Sensibilidad} = \frac{TP}{TP + FN}$$

$$\text{Especificidad} = \frac{TN}{TN + FP}$$

  En el ejemplo al que corresponde la matriz de la Figura 3.4, la sensibilidad valdría 5/8, y la especificidad, 3/5.

  Un clasificador será más eficiente cuanto mayores sean tanto la sensibilidad como la especificidad.

- **Precisión**: La precisión o PPV (*Positive Predictive Value*) mide la tasa de predicciones verdaderas. De forma análoga se define el indicador NPV (*Negative Predictive Value*).

$$\text{PPV} = \frac{TP}{TP + FP} \qquad \text{NPV} = \frac{TN}{TN + FN}$$

  En el ejemplo anterior vendrían dados por:

$$\text{PPV} = \frac{5}{5+2} = \frac{5}{7} \qquad \text{NPV} = \frac{3}{3+3} = \frac{1}{3}$$

- **F-measure**: Esta medida combina precisión y sensibilidad mediante el cálculo de la media armónica entre ambas. Viene dada por:

$$\text{F-measure} = \frac{2TP}{2TP + FP + FN}$$

  En el caso del ejemplo anterior, su valor resulta ser:

$$\text{F-measure} = \frac{2TP}{2TP + FP + FN} = \frac{10}{10+2+3} = \frac{10}{15}$$

- **Accuracy**: Es la tasa total de predicciones acertadas. Viene dada por la expresión:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$

  En el ejemplo anterior su valor sería:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} = \frac{5+3}{5+3+3+2} = \frac{8}{13}$$

  Este indicador de precisión no es útil cuando tenemos un problema de clasificación con dos clases no equilibradas. Supongamos que tenemos una clase A, con 1000 individuos, y una clase B, con 2 individuos. Si el clasificador siempre pronostica que los individuos a clasificar son de la clase A, la precisión es muy alta, 0.99, pero engañosa, pues nunca se detectaría un individuo de la clase B.

- **Análisis ROC**: Una curva ROC (del inglés, *Receiver Operating Characteristic*) es una representación gráfica de la **Sensibilidad** frente a **1-Especificidad** para un problema de clasificación binario. Identificando la **Sensibilidad** con el eje y y **1-Especificidad** con el eje x se tiene el espacio bidimensional conocido como espacio ROC donde la interpretación de las diferentes curvas ROC que

se representen en él nos proporciona una medida efectiva de evaluación de la precisión de un clasificador.

Para este tipo de análisis es necesario fijar un umbral o "threshold" en inglés, que no es más que un parámetro a partir del cuál los resultados se consideran positivos o negativos. Por ejemplo en una prueba diagnóstica, glicemia mayor de 125 mg/dl podría ser el umbral que separase "sanos"de "no sanos".

Fijado un clasificador, variando este parámetro umbral se van obteniendo las diferentes curvas ROC.

La medida de efectividad del clasificador radica en el estudio del área bajo la curva ROC (AUC, *Area Under the Curve*) (Figura 3.5).

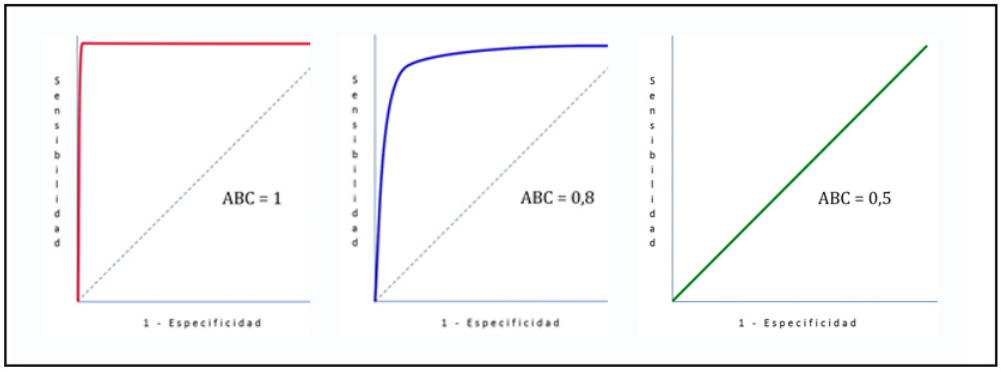

Figura 3.5: Diferentes ejemplos de curvas ROC.

Los puntos (0,0) y (1,1) corresponden, respectivamente, al clasificador que predice todo como clase positiva, y al clasificador que predice todo como clase negativa. El punto (0,1) corresponde al caso ideal, donde el clasificador acierta plenamente. Representar un clasificador de esta forma, permite que, dado un conjunto de clasificadores, se descarten aquellos que en el espacio ROC caigan dentro del cierre convexo de los puntos correspondientes a los distintos clasificadores, junto con los puntos (0,0), (1,0) y (1,1), ya que se encuentran dominados por los puntos de la frontera de dicho conjunto convexo. El mejor sistema de aprendizaje será aquel que produzca el conjunto de clasificadores con mayor área bajo la curva ROC

En una prueba diagnóstica, el AUC estima, por ejemplo, la capacidad de distinguir o de "discriminar" entre sanos y enfermos. Si el área bajo la curva valiese 1 la prueba sería perfecta, ya que clasificaría al 100 % de los enfermos como enfermos y al 100 % de los sanos como tales. En cambio, si el área bajo la curva valiese 0.5, existiría la misma probabilidad de clasificar a un enfermo como sano, o vicecersa,

que de acertar en la clasificación. Un área de 0.5 bajo la curva equivale a no discriminar, se interpreta como una prueba "no informativa".

- **Índice Kappa**: El índice o coeficiente Kappa es una medida de la concordancia del clasificador debida al azar, es decir, tiene en cuenta el hecho de que el clasificador haya acertado al clasificar un elemento en su clase correctamente por efecto del azar.

  - Valor 0: Concordancia perfecta
  - Valor 1: Concordancia debida al azar
  - Valor negativo: Concordancia menor de la que cabría esperar por el azar

  Si D es la suma de los elementos de la diagonal de la matriz de confusión, se define el índice Kappa como:

$$\kappa = \frac{D_{observado} - D_{azar}}{D_{perfecto} - D_{azar}}$$

Para la implementación de un clasificador es necesario tener en cuenta una serie de características concretas. La selección de éstas, sin embargo, no es fácil. La selección de características irrelevantes hace más compleja la tarea de clasificación, pues a medida que se va añadiendo más información, se incrementa la dimensión del espacio, y cada vez es más difícil conseguir un buen algoritmo de clasificación. Este fenómeno es de especial relevancia en el ámbito de la biología molecular, donde es frecuente que el número de características que se consideran sea superior al de individuos disponibles. En tales casos, resultan de indudable utilidad las **técnicas de selección de atributos o selección de características**, orientadas a identificar aquellas variables que realmente son relevantes para la clasificación, desechando aquellas otras con escaso poder discriminatorio o predictivo.

Capítulo 4

Técnicas de predicción de dianas de miRNA

El problema de la predicción de dianas de miRNA consiste principalmente en, dado un miRNA, identificar sobre que RNAm actúa. Este problema puede interpretarse como un problema de clasificación donde la muestra de aprendizaje estaría constituída por un conjunto de pares o dúplex formados por un miRNA y su RNAm diana, conocida previamente, y los elementos a clasificar serían RNAm candidatos a ser diana de un miRNA dado, de manera que las clases podrían interpretarse como "dúplex RNAm-miRNA válido" y "dúplex RNAm-miRNA no válido"(Figura 4.1).

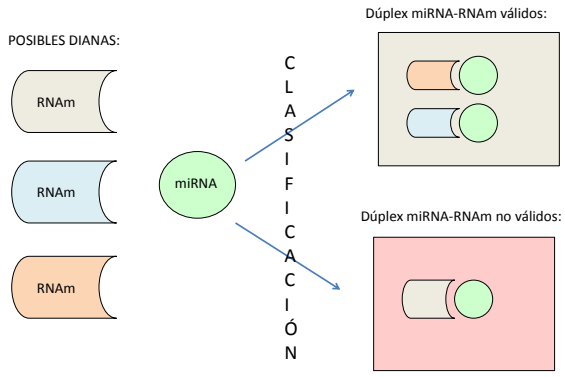

Figura 4.1: Proceso de clasificación de miRNA.

Debido a que la identificación experimental de dianas de miRNAs es un proceso complejo, se ha producido en los últimos tiempos un notable incremento en la predicción de dianas utilizando técnicas bioinformáticas, [43].

4.1. Estudio y selección de características relevantes

Un paso clave para la identificación de un RNAm diana es la selección de caracteres o características que se dan en la formación de un dúplex miRNA-RNAm y que son de gran poder predictivo. Con el esfuerzo de muchos investigadores y años de estudio se ha descubierto un número considerable de estas características.

En el 2003 empezó la primera ronda de predicción de dianas de miRNAs, descubriéndose por vía bioinformática que el miRNA *bantam* de la mosca, regula negativamente el gen proapoptótico *hid*. Otros grupos empezaron también a publicar predicciones de dianas de miRNAs en *D. melanogaster* y en vertebrados.

Una de las contribuciones más importantes para el reconocimiento de nuevas dianas fue el descubrimiento y *estudio de las regiones* Seed *y* outSeed, anteriormente descritas.

Las primeras predicciones se basaron en el estudio de la región de acoplamiento del miRNA que se veía en las dianas validadas experimentalmente, todos ellos localizados en la región 3' UTR. Los programas de predicción iniciales evaluaban el **grado de complementariedad existente entre la región *Seed* del miRNA y la región 3' UTR del RNAm diana**. Identificando dúplex miRNA-RNAm con un alto nivel de complementariedad, se llegaron a identificar más de 400 RNAms diana para miRNAs de vertebrados, [19, 23].

El estudio de la **Estabilidad termodinámica en los lugares de acoplamiento** ha sido fundamental pues se cree que el proceso de formación de un dúplex miRNA-RNAm está gobernado por un proceso de estabilidad termodinámica donde se llevan a cabo intercambios y variaciones de energía que se sospechan que pueden ayudar a detectar con precisión las dianas, [32].

Pequeñas diferencias en los algoritmos de predicción provocan gran diversidad de resultados en la predicción de dianas de miRNA. A parte de las diferencias en los algoritmos es importante tener en cuenta que no es sencillo definir la región 3' UTR de un gen. Las bases de datos existentes pueden diferir de forma considerable y también pueden no tener en cuenta muchos aspectos como la longitud de las regiones 3' UTR, las isoformas del tejido, el contenido nucleotídico, etc. Factores que pueden alterar claramente los resultados de los algoritmos que se desarrollen. Por ello, casi todos los algoritmos usan **sitios diana conservados evolutivamente**, como filtro biológico.

Ha sido considerado el hecho de que **varios miRNAs pueden actuar en concordancia para regular un mismo RNAm**. Esta idea ha sido corroborada por experimentos *in vitro*, así como por la observación de que **muchas regiones 3' UTR**

presentan sitios de unión para diferentes miRNAs. Se ha observado también en diversos casos que **múltiples sitios de unión para el mismo miRNA provocan un aumento de la represión** y esto ha sido incorporado en algunos de los algoritmos de predicción.

Otros factores que se han ido incluyendo en algunos algoritmos de predicción son el **grado de conservación entre especies, tanto de la región 3' UTR, como de los posibles sitios diana de miRNAs**.

El otro punto importante es la **integración de las anteriormente citadas relaciones regulatorias de los miRNAs en los datos genómicos funcionales conocidos**, así como también de **datos cuantitativos de los niveles de expresión de estos miRNAs en diferentes tejidos o periodos**. Todos estos últimos puntos, se han intentado también incluir en algunos algoritmos de predicción, de forma que hay una interacción cada vez mayor entre los datos experimentales que se van produciendo y los algoritmos de predicción.

4.2. Métodos computacionales desarrollados

Un aspecto importante relacionado con la predicción bioinformática de dianas de miRNA es la obtención de herramientas adecuadas para el almacenamiento de todos los datos posibles (nombre, contexto genómico, expresión, etc) acerca de los miRNA que se van descubriendo experimentalmente así como de sus correspondientes dianas, de manera que estén accesibles para toda la comunidad científica, [43].

En la tabla 4.2 se recoge un resumen de algunos de los algoritmos y bases de datos existentes que disponen de dianas de miRNA ya predichas en diferentes especies.

El algoritmo ***miRanda***, [30, 8, 19], es una de las herramientas más usadas. Dado un miRNA, *miRanda* genera un conjunto formado por todos los RNAm candidatos a ser diana y somete este conjunto a un proceso de filtrado. Para ello, usa tres parámetros de filtraje, de manera que se considerarán dianas válidas aquellas que superen estos filtros. Estos parámetros están asociados principalmente al nivel de complementariedad entre el miRNA y su posible diana y a la energía de enlace. Además, *miRanda* asocia a cada predicción una puntuación que describe de alguna manera la complementariedad entre las bases existente en el dúplex predicho. Por ejemplo, cada vez que aparece un apareamiento G:C se asigna +5 y cuando aparece A:T se asgina +2. Esta puntuación conocida como *miRanda Score* se computa como la suma de todas las puntuaciones que se den en un dúplex predicho. Así mismo, *miRanda* posee más de 25000 RNAm de humano conservados, por lo que a menudo resulta ser una base de datos muy útil.

BASE DE DATOS	DIRECCIÓN WEB	CARACTERÍSTICAS
miRBase	http://microrna.sanger.ac.uk/	Secuencias, dianas y registro
TargetScan	http://www.targetscan.org/	Provee dianas de miRNA conservadas en 5 vertebrados
TarBase	http://www.diana.pcbi.upenn.edu/tarbase.html	Dianas validadas experimentalmente
miRanda	http://www.microrna.org/	Posee dianas predichas
PiCTar	http://pictar.mdc-berlin.de	Proporciona detalles sobre las zonas 3' UTR de RNAm diana
RNA-hybrid	http://bibiserv.techfak.uni-bielefeld.de/mahybrid	Provee información sobre sitios de acomplamiento
TargetBoost	http://www.interagon.com/demo	Usa *machine learning* para detectar dianas
miRNAMap	http://mirnamap.mbc.nctu.edu.tw/	Se centra en la localización de las posibles dianas

Figura 4.2: Algunas bases de datos y algoritmos predictivos disponibles.

Otra herramienta muy útil es ***TarBase***, una base de datos que alberga una amplia colección de dianas de miRNA validados experimentalmente en humanos, ratón, mosca de la fruta, gusano, y pez cebra. Cada diana es descrita por el miRNA al que se une, el gen en el que se produce, la naturaleza de los experimentos que se realizaron para probarlo y la suficiencia de la unión para inducir la represión y/o escisión de la traducción. Además, la base de datos está funcionalmente ligada a varias otras bases de datos útiles como Gene Ontology (GO) y la UCSC Genome Browser.

miRBase funciona como libro de registro para todos y cada uno de los miRNA que van siendo descubiertos. Cada nuevo miRNA incorporado a la base de datos se designa con identificadores numéricos secuenciales. La base de datos utiliza prefijos de 3 o 4 letras para indicar de forma abreviada la especie en la que se ha identificado. Por ejemplo, para el humano el identificador es "hsa"(*Homo Sapiens*), de forma que si hablamos del miRNA *miR-101* debemos referirnos a él como *hsa-miR-101*.

Junto con la aparición de los algoritmos para la predicción de dianas de miRNA, surge la necesidad de evaluar la precisión y autenticidad de los resultados que proveen para seleccionar los que mejor precisión tengan. Para ello se tienen en cuenta los conceptos ya mencionados de *Sensibilidad* y *Especifidad*, ligados a las tasas de falsos

positivos y falsos negativos y por lo tanto describen de alguna manera el número de dianas deshechadas como falsas predicciones y el número de dianas que existen realmente pero que no son predichas computacionalmente. Por tanto, la elección de un algoritmo o programa para esta labor debería tener en cuenta estas dos tasas.

A continuación, se describen diferentes técnicas, basadas en algoritmos de clasificación, para la predicción de dianas de miRNA y la clasificación del estado de malignidad de un tumor según los perfiles de expresión de miRNA, así como una revisión del uso que se ha hecho de ellas en diferentes trabajos de investigación para el desarrollo de herramientas computacionales basadas en dichas técnicas, [43].

4.3. Clasificador Naïve Bayes

Comenzamos esta sección realizando una revisión de conceptos de cálculo de probabilidades. Consideremos para ello un espacio de probabilidad (Ω, A, P).

Probabilidad condicionada:

Nos indica la probabilidad de que acontezca un suceso cuando se tiene información a priori sobre otro suceso.

Sea B un suceso de probabilidad no nula, $P(B) > 0$. Para cualquier otro suceso A, se llama probabilidad condicionada de A a B a:

$$P(A/B) = \frac{P(A \cap B)}{P(B)}$$

Dos sucesos se dicen **independientes** si $P(A \cap B) = P(A)P(B)$ o, equivalentemente, $P(A/B) = P(A)$.

Sistema completo de sucesos:

Se denomina sistema completo de sucesos a un conjunto de sucesos $\{A_1, A_2, .., A_r\}$, mutuamente excluyentes, de manera que su unión proporciona el espacio muestral Ω. Esto es:

$$\cup_{i=1}^{r} A_i = \Omega \qquad\qquad A_i \cap A_j = \emptyset \quad \forall\, i, j, \text{ tal que } i \neq j$$

Teorema de la probabilidad total:

Sea $\{A_1, A_2, .., A_r\}$ un sistema completo de sucesos tales que $P(A_i) > 0$ para todo i.

Entonces, para cualquier otro suceso B se tiene :

$$P(B) = \sum_{i=1}^{r} P(B/A_i)P(A_i)$$

Teorema de Bayes:

Sea $\{A_1, A_2, A_3, ..., A_r\}$ un sistema completo de sucesos tal que $P(A_i) > 0$ para todo i. Si B es otro suceso tal que $P(\mathrm{B}) > 0$, entonces, para cualquier índice $j \in \{1, \ldots, r\}$ se verifica:

$$P(A_j/B) = \frac{P(B/A_j)P(A_j)}{P(B)}$$

donde $P(B) = \sum_{i=1}^{r} P(B/A_i)P(A_i)$ viene dada por el Teorema de la Probabilidad Total.

Naïve Bayes, [22, 40, 25], es un clasificador basado en el cálculo de la probabilidad de que un elemento dado pertenezca a cada una de las clases consideradas haciendo uso del Teorema de Bayes.

El principal objetivo consiste en determinar a qué clase dentro del conjunto $C = (c_1, c_2, .., c_m)$ pertenece un elemento descrito por un vector de características $x = (x_1, \ldots, x_n)$.

El procedimiento Naïve Bayes se basa en, conocido el vector de características, calcular la probabilidad de pertenencia a cada una de las clases $P(c_i/x)$ con $i = 1, ..., m$ y clasificar al elemento en la clase que proporcione máxima probabilidad.

Sea c^* la clase que proporciona la máxima probabilidad, y por lo tanto en la que se clasifica el elemento.

$$c^* = \arg \max_{i=1,...,m} P(c_i/x)$$

La probabilidad anterior se calcula haciendo uso del Teorema de Bayes, de manera que el problema a resolver se reduce a:

$$c^* = \arg \max_{i=1,...,m} \frac{P(x/c_i)P(c_i)}{P(x)} = \arg \max_{i=1,...,m} P(x/c_i)P(c_i)$$

El inconveniente más importante que, en principio, presenta este clasificador es el alto coste computacional que conlleva, debido a la gran cantidad de probabilidades que

hay que estimar a partir de la muestra de aprendizaje. Así, por ejemplo, si suponemos que las variables x_i son dicotómicas, entonces el número de probabilidades a estimar sería: $m-1+m(2^n-1)$ donde $m-1$ corresponden a $P(c_i)$ y $m(2^n-1)$ a $\mathrm{P}(x/c_i)$.

En la Tabla 4.1 se muestra el número de parámetros a estimar según diferentes valores de m y n en un problema binario; como puede apreciarse, este número es muy elevado incluso para valores reducidos de los parámetros m y n.

El clasificador Naïve Bayes reduce considerablemente el coste computacional introduciendo una hipótesis previa en la formulación del problema. Esta hipótesis "naïve" consiste en suponer que, conocida la clase a la que pertenece un individuo, las variables x_i del vector de características x que describe al individuo son **independientes** entre sí. Por tanto, bajo esta hipótesis el problema a resolver se reduce a:

$$c^* = \arg \max_{i=1,\ldots,m} P(x_1/c_i)P(x_2/c_i)P(x_3/c_i)...P(x_n/c_i)P(c_i)$$

Así, el coste computacional en el caso de variables dicotómicas se reduce a $m-1+mp$, sensiblemente menor que en el caso general, como se puede observar en la Tabla 4.2.

m	n	parámetros
4	10	4095
5	20	5.242.879
10	30	10.737.418.239

Tabla 4.1: Número de parámetros a estimar según número de clases y variables

A pesar de que la hipótesis de independencia en las variables predictoras es muy fuerte y se cumple en pocas ocasiones, el clasificador Naïve Bayes se emplea en numerosos campos y proporciona, en general, buenos resultados.

m	n	parámetros
4	10	43
5	20	104
10	30	309

Tabla 4.2: Número de parámetros a estimar bajo la hipótesis naïve.

4.3.1. Estimación de los parámetros

Pasemos ahora a estudiar cómo se estiman las probabilidades utilizadas en el clasificador Naïve Bayes.

Todos los parámetros que aparecen en la expresión del clasificador Naïve Bayes usualmente se estiman a partir de la muestra de aprendizaje:

- La estimación de las probabilidades $P(c_i)$ se realiza dividiendo el número de individuos que pertenecen a la clase c_i entre el número total de individuos en la muestra de aprendizaje.

- Para la estimación de las probabilidades $P(x_j/c_i)$ se consideran sólo los individuos pertenecientes a la clase c_i y se divide el número de ellos que presentan la característica x_j entre el número total de individuos de la clase c_i. Este procedimiento es válido si las variables x_j son cualitativas o cuantitativas discretas, aunque también es válido para el caso de variables continuas, usando en este caso las funciones de densidad correspondientes, como se describe a continuación.

El desarrollo de la técnica Naïve Bayes presentado hasta ahora solo es válido si las variables x_j son cualitativas o cuantitativas discretas. En el caso de variables continuas, el problema a resolver resulta ser:

$$c^* = \arg \max_{i=1,\dots,m} f_{x_1/c_i}(x_1) f_{x_2/c_i}(x_2) \dots f_{x_n/c_i}(x_n) P(c_i)$$

donde f_{x_j/c_i} representa la función de densidad de la variable x_j sobre la clase c_i. En muchas ocasiones se asume que estas variables siguen una distribución Normal (u otra que

se considere adecuada), estimándose sus parámetros a partir de la información disponible en la muestra de aprendizaje. Alternativamente, puede realizarse una estimación no paramétrica mediante funciones *Kernel*, [44, 48].

4.3.2. Ejemplo de aplicación

Para una mejor interpretación de esta técnica de clasificación, se presenta a continuación un pequeño ejemplo ilustrativo. Se pretende realizar la clasificación de terrenos en terrenos *cultivables* o *no cultivables*, en función de las siguientes características:

- x_1: Temperatura media (Muy Baja/Baja/Media/Alta)
- x_2: Existencia de agua cercana (Sí/No)
- x_3: Localización (Valle/Llanura/Alta montaña/Desierto)
- x_4: Extensión (Pequeña/Grande)
- x_5: Cercano a población (Sí/No)

Las clases en las que se pretende clasificar un nuevo terreno con un vector de características determinado son: c_1="Terreno cultivable" y c_2="Terreno no cultivable". Se dispone de una muestra de aprendizaje, cuya descripción aparece recogida en la Tabla 4.3.

Terrenos	Temperatura media	Fuente de agua cercana	Localización	Extensión	Cercano a población	Terreno cultivable
1	Alta	Sí	Llanura	Grande	Sí	Sí
2	Alta	No	Valle	Grande	No	No
3	Baja	Sí	Valle	Pequeña	Sí	Sí
4	Media	Sí	Llanura	Grande	Sí	Sí
5	Muy baja	Sí	Alta montaña	Pequeña	No	No
6	Media	Sí	Llanura	Pequeña	No	Sí
7	Media	No	Desierto	Grande	No	No

Tabla 4.3: Ejemplo de aprendizaje utilizando Naïve Bayes.

Dado un terreno con el siguiente vector de características

$$x = (x_1 = \text{“Media”}, x_2 = \text{“Sí”}, x_3 = \text{“Valle”}, x_4 = \text{“Grande”}, x_5 = \text{“No”})$$

se pretende determinar si dicho terreno es o no cultivable empleando para ello el clasificador Naïve Bayes.

Realizando los cálculos oportunos, se tiene:

$$P(c_1) = \frac{4}{7} \qquad P(c_2) = \frac{3}{7}$$

$$P(x_1 = \text{“Media”}/c_1) \times P(x_2 = \text{“Sí”}/c_1) \times P(x_3 = \text{“Valle”}/c_1) \times$$
$$\times P(x_4 = \text{“Grande”}/c_1) \times P(x_5 = \text{“No”}/c_1) \times P(c_1) = 0.0141$$

$$P(x_1 = \text{“Media”}/c_2) \times P(x_2 = \text{“Sí”}/c_2) \times P(x_3 = \text{“Valle”}/c_2) \times$$
$$\times P(x_4 = \text{“Grande”}/c_2) \times P(x_5 = \text{“No”}/c_2) \times P(c_2) = 0.0067$$

Como la probabilidad es mayor en el primer caso, clasificamos el terreno como *cultivable*.

4.3.3. Aplicación en la predicción de dianas de miRNA

Se describe a continuación la aplicación del clasificador Naïve Bayes al problema de la predicción de dianas de miRNA. Recientemente ha sido desarrollado el algoritmo **NBmiRtar**, basado en este clasificador, [61]. A continuación se presentan los diferentes pasos para la construcción de esta herramienta así como los resultados que ha proporcionado.

Datos para el desarrollo del algoritmo

Para el desarrollo de este algoritmo se ha usado una colección de 225 RNAm dianas de miRNA confirmados de diversas especies (humano, ratón, mosca de la fruta, gusano y pez cebra). Además, se ha usado un conjunto de 38 RNAm que se sabe que no son diana, es decir, un conjunto de falsas predicciones. Este conjunto de RNAm se han obtenido de la base de datos reseñada anteriormente *TarBase*, [50]. Ocurre que el conjunto de ejemplos negativos es muy pequeño en comparación con el conjunto de ejemplos positivos y esto podría dar lugar a un clasificador con un comportamiento sesgado, ya

que tendría dificultad para identificar los ejemplos negativos. Este problema se resuelve mediante la generación de ejemplos negativos de forma artificial. Tras este proceso de generación de ejemplos negativos artificiales se tendría constituida la **muestra de aprendizaje**, donde cada ejemplo vendría constituido por un dúplex miRNA-RNAm con su **clase correspondiente**, es decir, "dúplex válido" o "dúplex no válido".

Para la generación de ejemplos negativos de forma artificial se hace uso de la herramienta *MiRanda*, [30, 8, 19].

Generación de ejemplos negativos artificiales

Para generar el conjunto artificial de ejemplos negativos se ha usado una colección de 3000 miRNA artificiales de 30 nucleótidos de longitud cada uno. Estos miRNA artificiales consisten en una cadena aleatoria de nucleótidos con frecuencias de aparición de las bases A, C, G y U de 0.34, 0.19, 0.18 y 0.29, respectivamente, frecuencias que son muy diferentes de que las que presentan las bases en los miRNA reales.

Se ha usado la herramienta *miRanda* para, de los 29785 RNAm de humanos que tiene conservados, encontrar predicciones para esos 3000. Para ello, se fijan los parámetros filtro de *miRanda*: energía de enlace a 25 kCAL/mol y *MiRanda* Score a 180. De esta forma se predicen 133316 dianas falsas, que se toman como conjunto de ejemplos negativos.

Diseño de la estructura de un dúplex miRNA-RNAm y selección de características relevantes

Este algoritmo se basa principalmente en el estudio de las regiones *Seed* y *outSeed*. Para el diseño de esta herramienta, se parte de que las siguientes hipótesis deben cumplirse para tener un dúplex miRNA;RNAm funcional, [9, 57, 24]:

- Para la formación de un dúplex funcional es necesario una complementariedad de entre 7 u 8 bases en la región *Seed*.
- Una complementariedad débil en la región *Seed* puede verse compensada por una buena complementariedad en la región *outSeed*.
- Una buena complementariedad en la región *outSeed* por sí sola no es suficiente para asegurar la formación de un dúplex funcional.

La selección de las características relevantes para el proceso de clasificación es una tarea compleja, pues si, por ejemplo, solo se tienen en cuenta características focalizadas en la región *Seed*, el algoritmo no será capaz de estudiar los casos en los que la complementariedad se encuentra compensada en la región *outSeed*. Por ello, cada dúplex miRNA-RNAm se particiona según estas dos regiones (Figura 4.3). De estas dos regiones se toman las características consideradas en el algoritmo.

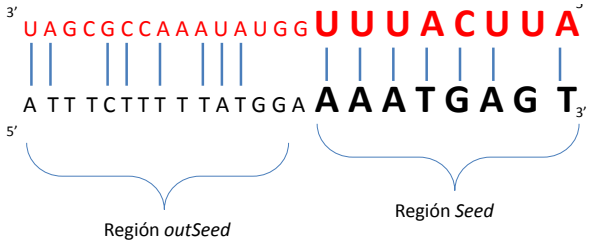

Figura 4.3: miRNA hsa-miR-579 y su diana LRIG3 particionado en las dos regiones.

El **conjunto de características** para este algoritmo está formada por 57 características estructurales de ambas regiones como pueden ser el número de bases emparejadas, el número de bucles existentes, distancia entre ciertas parejas de bases, etc., [62].

Esquema, entrenamiento y validación de NBmiRTar

El clasificador se aplica a la salida inmediata de *miRanda* (Figura 4.4). Esta herramienta genera un gran número de predicciones, de manera que su posterior tratamiento y análisis conlleva un alto coste. Para reducir el número de elementos de salida, se seleccionan únicamente las predicciones que proporciona *miRanda* con un nivel de energía de enlace, según el MFE (*Minimum Free Energy*), menor o igual a 12kCal/mol y un *miRanda Score* de 90, [28].

El interés principal es construir un clasificador con alta especificidad a la hora de identificar dianas para reducir la tasa de falsos positivos a niveles tales que hagan viable la realización de procesos de validación en laboratorio. Incluso con un porcentaje pequeño de falsos negativos se generarían miles de predicciones, dificultando considerablemente el proceso de validación. Por tanto, el punto clave es definir el parámetro umbral o *threshold* que consiga eliminar cuantos más falsos positivos sea posible.

Para esta tarea, se incluye una puntuación NB propia del clasificador que se asigna a cada dúplex candidato a ser funcional y según su valor se clasifica en una de las

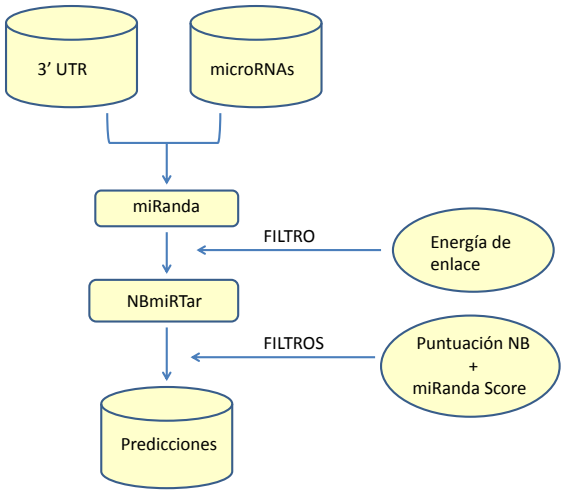

Figura 4.4: Esquema de funcionamiento de NBmiRTar.

dos clases posibles tomando como referencia un valor umbral. Estudiando las tasas de falsos positivos y de falsos negativos que se dan según se varía el parámetro umbral, se llega a la conclusión de que un parámetro umbral de 0.9 reduce la tasa de falsos positivos en un 64 % perdiendo únicamente un 4.5 % de las predicciones verdaderas.

El entrenamiento del algoritmo se realiza mediante un proceso de validación cruzada con 10 pliegues. En cada iteración, se selecciona al azar un 90 % de cada clase para conjunto de entrenamiento y el resto queda para determinar el rendimiento del algoritmo. Para determinar la media de predicciones de verdaderos positivos y verdaderos negativos se repite este proceso 100 veces.

Se ha evaluado la precisión de NBmiRTar sobre el conjunto de ejemplos positivos y el conjunto de ejemplos negativos generados artificialmente. Con el conjunto de ejemplos positivos se ha obtenido una sensibilidad de 0.93 y una especificidad de 0.7, mientras que usando 900 ejemplos negativos la sensibilidad fue 0.94 y la especificidad 0.99.

La capacidad de NBmiRTar para mantener altas las tasas de sensibilidad y especificidad sugiere que, efectivamente, las características estructurales de ambas regiones *Seed* y *outSeed* que se han escogido para el desarrollo del algoritmo caracterizan un dúplex funcional, así como que el conjunto de ejemplos negativos generados artificialmente parce ser una representación precisa de la realidad.

4.4. Clasificador Support Vector Machine

Dentro de la clasificación, las Máquinas de Vectores Soporte, o SVMs (del inglés, *Support Vector Machine*), pertenecen a la categoría de los clasificadores lineales, puesto que inducen separadores lineales o hiperplanos, ya sea en el espacio original de los ejemplos de la muestra de aprendizaje, si éstos son separables o cuasi-separables (ruido), o en un espacio transformado si los ejemplos no son separables linealmente en el espacio original, [16]. Mientras la mayoría de los métodos de aprendizaje se centran en minimizar los errores cometidos por el modelo generado a partir de los ejemplos de entrenamiento (error empírico), las SVMs se centran en la minimización del riesgo estructural. Aunque originariamente las SVMs fueron pensadas para resolver problemas de clasificación binaria, actualmente se utilizan para resolver otros tipos de problemas (regresión, agrupamiento, multiclasificación, etc.), [29, 7, 18, 54, 55].

4.4.1. SVM para clasificación binaria de elementos linealmente separables

Consideremos una muestra de aprendizaje T dada por:

$$T = \{(x_1, y_1), (x_2, y_2), ..., (x_p, y_p)\}$$

donde $x_i = (x_{i1}, x_{i2}, ..., x_{in})$ con $x_i \in \mathbb{R}^n$ e $y_i \in \{-1, +1\}$ para $i = 1, \ldots, p$

Un **hiperplano de separación** puede definirse como una función afín capaz de separar los elementos del conjunto T sin error (Figura 4.5). Vendría dado por:

$$h(x) = (\omega_1 x_1 + \omega_2 x_2 + ... + \omega_n x_n) + b =< \omega, x > +b$$

donde $\omega = (\omega_1, \omega_2, ..., \omega_n)$ es el vector ortogonal al hiperplano, $b \in \mathbb{R}$ y $<,>$ expresa el producto escalar habitual.

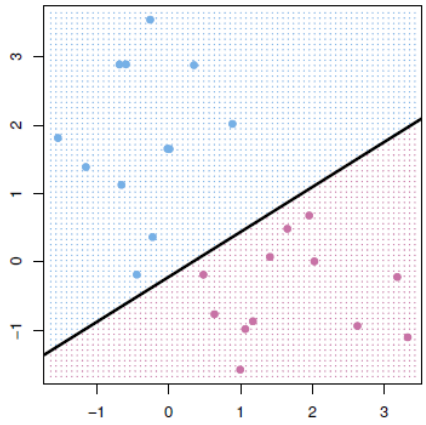

Figura 4.5: Hiperplano de separación en un espacio bidimensional con elementos separables en dos clases.

El hiperplano de separación $h(x)$ cumplirá las siguientes restricciones para todo elemento x_i, $i = 1, \ldots, n$ de la muestra de aprendizaje:

$$\begin{aligned} h(x_i) &=< \omega, x_i > +b > 0 \qquad \text{si } y_i = +1 \\ h(x_i) &=< \omega, x_i > +b < 0 \qquad \text{si } y_i = -1 \end{aligned}$$

o, de forma más concreta:

$$y_i h(x_i) > 0 \qquad i = 1, ..., n$$

Tal y como puede deducirse de la Figura 4.5, el hiperplano de separación no es único, existiendo una infinidad de hiperplanos que cumplen las restricciones anteriores.

Todos los hiperplanos de la Figura 4.6 en principio generan clasificadores igualmente válidos sobre la muestra de aprendizaje, ya que todos clasifican correctamente la totalidad de la misma. Sin embargo, no todos son igualmente efectivos cuando actúan sobre elementos que no han formado parte del aprendizaje. Por ejemplo, si se utiliza el clasificador dado por el hiperplano de separación rojo, el * sería clasificado en la clase rojo, cuando parece evidente que * pertenece a la clase azul. Esto, sin embargo, no ocurriría con los otros dos clasificadores. Es entonces cuando surge la pregunta de si es posible establecer algún criterio adicional para definir un hiperplano de separación **óptimo**, en el sentido de que se reduzca al mínimo la posibilidad de ocurrencia de este fenómeno.

Para ello, se define lo que se conoce como **margen de un hiperplano de separación**, que no es más que la distancia entre dicho hiperplano y el ejemplo más cercano

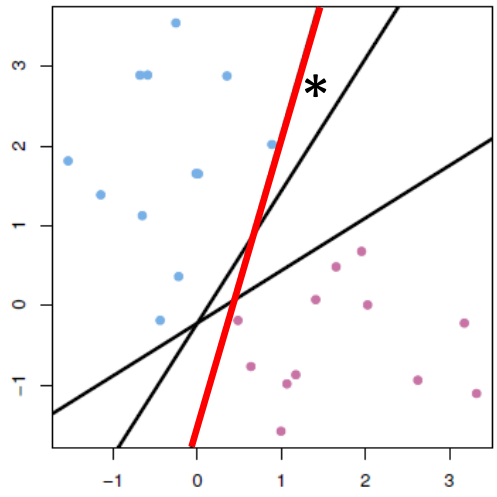

Figura 4.6: Diferentes hiperplanos separadores.

de cualquiera de las dos clases. Se denota por τ. Teniendo en cuenta esta definición, se considerará como hiperplano de separación óptimo aquel que consiga el máximo margen de separación.

SVM de margen máximo

La distancia entre un hiperplano $h(x) =< \omega, x > +b$ y un punto x' viene dada por:

$$\frac{|h(x')|}{||\omega||}$$

siendo $|.|$ el operador valor absoluto y $||.||$ el operador norma de un vector.

Según la expresión anterior, si consideramos un hiperplano de separación, su margen vendrá dado por:

$$\tau = \min_{x \in T} \frac{|h(x)|}{||\omega||} \tag{4.1}$$

En caso de ser $h(x)$ un hiperplano de separación, se cumple $y_i h(x_i) > 0$ para $i = 1, ..., n$, de donde se deduce que todos los elementos de la muestra de aprendizaje cumplirán:

$$\frac{y_i h(x_i)}{||\omega||} > \tau \qquad \text{ó bien} \qquad y_i h(x_i) > \tau||\omega||$$

De la expresión (4.1) se deduce que encontrar el hiperplano óptimo es equivalente a encontrar el vector ω que maximiza el margen. Sin embargo, existen infinitas soluciones que difieren en la escala de ω; por ejemplo, todas las funciones afines de la forma $\lambda(< \omega, x > +b)$ con $\lambda \in \mathbb{R}$, representan el mismo hiperplano. La homogeneidad de la expresión anterior permite imponer la condición $\tau||\omega|| = 1$, llegando a la conclusión de que aumentar el margen es equivalente a disminuir la norma del vector ω, [12].

Por tanto, de acuerdo a su definición, un hiperplano de separación óptimo será aquel que posea un **margen máximo** y, por tanto, un valor mínimo de $||\omega||$ y además cumpla

$$y_i h(x_i) \geq 1 \qquad i = 1, ..., n.$$

Los ejemplos que se encuentran situados a ambos lados del hiperplano y que definen el margen, es decir, $y_i h(x_i) = 1$ reciben el nombre de **vectores soporte**.

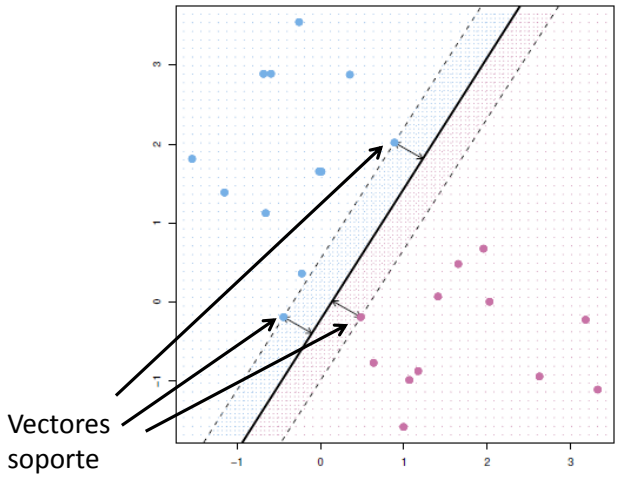

Figura 4.7: Vectores soporte.

Determinación del hiperplano de margen máximo

La búsqueda del hiperplano óptimo para el caso separable puede ser formalizada como el problema de encontrar los valores de ω y b que minimizan $||\omega||$, clasificando correctamente todos los ejemplos, lo que equivale a resolver el problema de optimización:

$$\begin{array}{ll} \text{mín} & \frac{1}{2} < \omega, \omega > \\ \text{s.a.} & y_i(\omega^t x_i + b) - 1 \geq 0 \qquad i = 1, ..., p \\ & \omega \in \mathbb{R}^n, b \in \mathbb{R} \end{array}$$

Se trata, por lo tanto, de un problema convexo con restricciones lineales. La teoría de optimización establece que un problema de optimización, denominado primal, tiene una forma dual si la función a optimizar y las restricciones son funciones estrictamente convexas. En estas circunstancias, resolver el problema dual permite obtener la solución del problema primal.

Así pues, el problema anterior satisface el criterio de convexidad y, por lo tanto, tiene un dual. En estas condiciones, se pueden enumerar los siguientes pasos encaminados a resolver el problema primal:

Primeramente, se construye un problema de optimización sin restricciones utilizando la función Lagrangiana:

$$L(\omega, b, \alpha) = \frac{1}{2} < \omega, \omega > - \sum_{i=1}^{p} \alpha_i [y_i(< \omega, x_i > +b) - 1] \tag{4.2}$$

donde α_i son los denominados *multiplicadores de Lagrange*.

El segundo paso consiste en aplicar las condiciones de Karush-Kuhn-Tucker, también conocidas como condiciones KKT [59, 36]:

$$\frac{\partial L(\omega, b, \alpha)}{\partial \omega} - \omega - \sum_{i=1}^{p} \alpha_i y_i x_i = 0 \qquad i = 1, \ldots, p \tag{4.3}$$

$$\frac{\partial L(\omega, b, \alpha)}{\partial \omega} - \sum_{i=1}^{p} \alpha_i y_i = 0 \qquad i = 1, \ldots, p \tag{4.4}$$

$$\alpha_i [1 - y_i(< \omega, x_i > +b)] = 0 \qquad i = 1, \ldots, p \tag{4.5}$$

Las restricciones dadas por (4.3)-(4.4) corresponden al resultado de aplicar la primera condición KKT (gradiente de la función lagrangiana igual a cero) y la expresada en (4.5) corresponde al resultado de aplicar la segunda condición KKT (condición de complementariedad). Las primeras permiten expresar los parámetros de ω y b en términos de α_i:

$$\omega = \sum_{i=1}^{p} \alpha_i y_i x_i \qquad i = 1, \ldots, p \tag{4.6}$$

y, además, establecen restricciones adicionales para los coeficientes α_i:

$$\sum_{i=1}^{p} \alpha_i y_i = 0 \qquad i = 1, \ldots, p \tag{4.7}$$

Con las nuevas relaciones obtenidas se construirá el problema dual. Así, bastará usar (4.6) para expresar la función Lagrangiana sólo en términos de α_i. Antes de ello, se puede reescribir (4.2) como

$$L(\omega, b, \alpha) = \frac{1}{2} < \omega, \omega > - \sum_{i=1}^{p} \alpha_i y_i - b \sum_{i=1}^{p} \alpha_i y_i + \sum_{i=1}^{p} \alpha_i$$

Teniendo en cuenta que, según la condición (4.7), el tercer término de la expresión anterior es nulo, la sustitución de (4.6) en dicha expresión proporciona:

$$\begin{aligned} L(\alpha) &= \frac{1}{2}\big(\sum_{1=1}^{p} \alpha_i y_i x_i\big)\big(\sum_{j=1}^{p} \alpha_j y_j x_j\big) - \big(\sum_{i=1}^{p} \alpha_i y_i x_i\big)\big(\sum_{j=1}^{p} \alpha_j y_j x_j\big) + \sum_{i=1}^{p} \alpha_i \\ L(\alpha) &= -\frac{1}{2}\big(\sum_{i=1}^{p} \alpha_i y_i x_i\big)\big(\sum_{j=1}^{p} \alpha_j y_j x_j\big) + \sum_{i=1}^{p} \alpha_i \\ L(\alpha) &= \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \alpha_i \alpha_j y_i y_j < x_i, x_j > \end{aligned} \tag{4.8}$$

Es decir, hemos transformado el problema de minimización primal en el problema dual consistente en maximizar (4.8) sujeto a las restricciones anteriores. Es decir, el problema dual resulta ser:

$$\begin{aligned} \text{máx} \quad & L(\alpha) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{p} \alpha_i \alpha_j y_i y_j < x_i, x_j > \\ \text{s.a.} \quad & \sum_{i=1}^{p} \alpha_i y_i = 0 \\ & \alpha_i \geq 0 \qquad i = 1, \ldots, p \end{aligned}$$

Al igual que el problema primal, este problema es abordable mediante técnicas estándar de programación cuadrática. Sin embargo, como se puede comprobar, el tamaño del problema de optimización dual crece con el número de ejemplos de la muestra de aprendizaje, p, mientras que el problema primal lo hace con la dimensionalidad, n. Por tanto, aquí radica la ventaja del problema dual, es decir, el coste computacional asociado a su resolución es factible incluso para problemas en dimensión muy alta.

La solución del problema dual, α^*, nos permitirá obtener la solución del sistema primal. Para ello bastará sustituir dicha solución en la expresión (4.6) y sustituir el resultado obtenido en (4.2), es decir,

$$D(x) = \sum_{i=1}^{n} \alpha_i^* y_i < x, x_i > + b^* \tag{4.9}$$

Volviendo a las restricciones (4.8), se puede afirmar que si $\alpha_i > 0$ entonces el segundo factor de la parte izquierda de dicha expresión tendrá que ser cero y, por tanto

$$y_i(< \omega^*, x_i > + b^*) = 1 \tag{4.10}$$

Es decir, el correspondiente ejemplo (x_i, y_i) satisface en igualdad la restricción asociada del problema primal. Por definición, los ejemplos que satisfacen las restricciones dadas por (4.10) son los vectores soporte y, por consiguiente, se puede afirmar que sólo los ejemplos que tengan asociado un multiplicador $\alpha_i > 0$ serán vectores soporte. De este resultado, también puede afirmarse que el hiperplano de separación (4.9) se construirá como una combinación lineal de sólo los vectores soporte de la muestra de aprendizaje, ya que el resto de ejemplos tendrán asociado un multiplicador $\alpha_j = 0$.

Para que la definición del hiperplano (4.9) sea completa es preciso determinar el valor del parámetro b^*. Su valor se obteniendo despejando b^* en (4.10).

$$b^* = y_{vs} - < \omega^*, x_{vs} > \tag{4.11}$$

donde (x_{vs}, y_{vs}) representa cualquier vector soporte junto con su etiqueta de clase, es decir, la tupla de cualquier ejemplo que tenga asociado un multiplicador $\alpha_i \neq 0$. En la práctica, es más robusto obtener el valor de b^* promediando a partir de todos los vectores soporte, N_{vs}. Así, la expresión (4.11) se transforma en

$$b^* = \frac{1}{N_{vs}} \sum_{i=1}^{N_{vs}} (y_{vs} - < \omega^*, x_{vs} >) \tag{4.12}$$

4.4.2. SVM para clasificación binaria de elementos cuasi-separables linealmente

En muchas ocasiones, las clases que intervienen en el problema de clasificación no son separables linealmente, como es el caso representado en la Figura 4.8. En otras, a pesar de contar con clases linealmente separables, es frecuente encontrar observaciones *outliers* que dan lugar a un hiperplano de separación no adecuado. Así, por ejemplo, en la Figura 4.9 se aprecia que la observación marcada con * va a ser incorrectamente clasificada debido a la existencia de una observación *outlier* que distorsiona la posición óptima del hiperplano de separación.

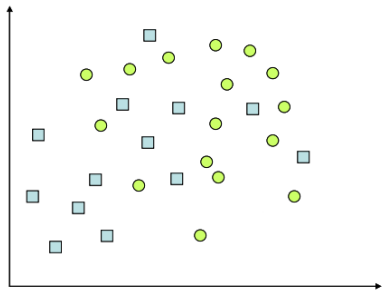

Figura 4.8: Clases no separables linealmente.

La estrategia para este tipo de problemas es "relajar" el grado de separabilidad de los elementos de la muestra de aprendizaje, permitiendo que algunos elementos no queden clasificados correctamente. La idea para abordar este problema es introducir en la formulación del problema de la determinación del hiperplano óptimo, un conjunto de **variables de holgura**, ψ_i con $i = 1, ..., n$ que permitirán cuantificar el número de observaciones mal clasificadas que se está dispuesto a admitir, es decir,

$$y_i(< \omega, x_i > +b) \geq 1 - \psi_i \qquad \psi_i \geq 0\,,\; i = 1, \ldots, p$$

Por tanto, para un ejemplo (x_i, y_i), su variable de holgura ψ_i representa la desviación del caso separable, medida desde el borde del margen que corresponde a la clase y_i (Figura 4.10).
La suma de todas las variables de holgura , $\sum_{i=1}^{p} \psi_i$, permite medir la importancia o el impacto asociado al número de ejemplos no separables. Generalmente, aunque depende de la magnitud de las ψ_i, cuanto mayor sea el valor de la suma anterior, mayor será el número de ejemplos no separables.

La perturbación que produce en el problema de optimización introducir las variables

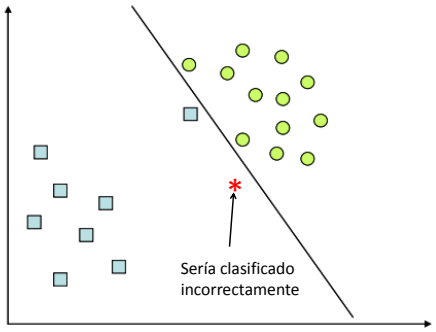

Figura 4.9: Influencia de los *outliers* en SVM.

de holgura (Figura 4.10) se controla por medio de una ***penalización C*** en la función objetivo, teniéndose entonces el siguiente problema de optimización:

$$\begin{aligned} \text{mín} \quad & \omega^t\omega + C\sum_{i=1}^{p}\psi_i & & \\ \text{s.a.} \quad & y_i(\omega^t x_i + b) \geq 1 - \psi_i & & i = 1,\ldots,p \\ & \psi_i \geq 0 & & i = 1,\ldots,p \\ & \omega \in \mathbb{R}^n\,,\, b \in \mathbb{R} & & \end{aligned} \tag{4.13}$$

La constante C permite controlar la influencia de los ejemplos no separables. Así, un valor muy grande de C obligará a que las variables de holgura ψ_i tomen valores pequeños e incluso nulos. En el límite ($C \to \infty$) tendríamos el problema del margen máximo, pues estaríamos obligando a que $\psi_i = 0$. Por contra, un valor de C pequeño permitiría valores de ψ_i grandes, por lo que se estaría admitiendo un número amplio de ejemplos mal clasificados. En el otro caso límite ($C \to 0$), se permitiría que todos los elementos resultaran mal clasificados, pues ψ_i podría tomar valores arbitrariamente grandes.

Como en el caso de la sección anterior, el problema de optimización (4.13) puede ser llevado a su forma dual para facilitar su resolución. Entonces:

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{i=1}^{p} \alpha_i \alpha_j y_j x_i^t x_j \\ \text{s.a.} \quad & \sum_{i=1}^{p} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \qquad i = 1, \ldots, p \end{aligned} \tag{4.14}$$

El hiperplano así definido recibe el nombre de hiperplano de separación con **margen débil**.

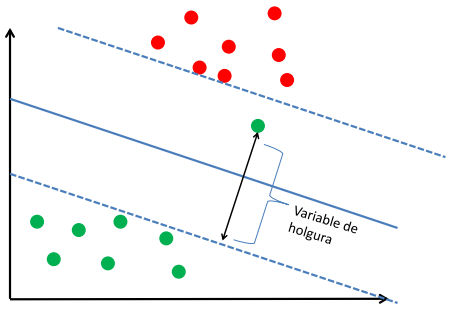

Figura 4.10: Medida de la variable de holgura.

4.4.3. SVM para clasificación binaria de elementos no separables linealmente

En ocasiones el SVM lineal no proporcionará una buena clasificación aunque se emplee el SVM con margen débil, con independencia del valor de la constante de regularización C. La estrategia común en estos casos consiste en realizar una inmersión de los elementos de la muestra de aprendizaje en un espacio de dimensión mayor, donde sí sean linealmente separables (Figura 4.11).

Para llevar a cabo esta inmersión de los datos en un espacio F de dimensionalidad mayor, necesitamos una función de inmersión. En el caso de la Figura 4.11 esa función de inmersión sería la parábola $y = x^2$. A esta función se la denomina función de inmersión base.

Sea $\phi : \mathbb{X} \subset \mathbb{R}^n \rightarrow F$ la función de inmersión que hace corresponder a cada vector en el espacio de entrada $\mathbb{X}$ un punto en el espacio transformado F donde:

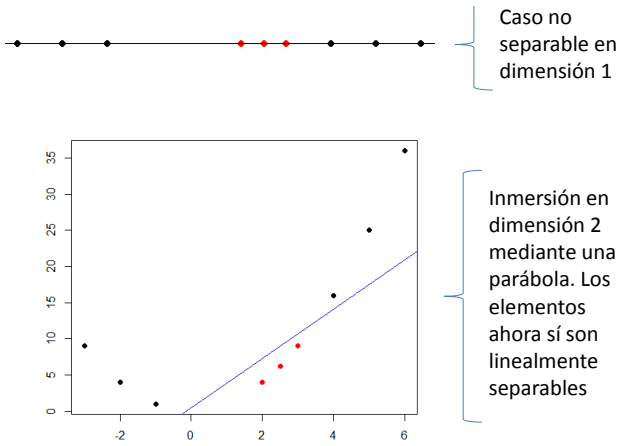

Figura 4.11: Inmersión en un espacio de dimensión mayor.

$$\phi(x) = (\phi_1(x), \phi_2(x), ..., \phi_m(x))$$

donde al menos una función $\phi_i(x)$ es una función no lineal. La idea entonces es construir un hiperplano de separación lineal en este nuevo espacio, dado por:

$$h(x) = (\omega_1\phi_1(x), \omega_2\phi_2(x), ..., \omega_m\phi_m(x)) =< \omega, \phi(x) >$$

El problema de optimización a resolver quedaría, por tanto:

$$\begin{aligned} \text{mín} \quad & \omega^t\omega \\ \text{s.a.} \quad & y_i(\omega^t\phi(x_i) + b) \geq 1 \qquad i = 1, \ldots, p \\ & \omega \in F, b \in \mathbb{R} \end{aligned}$$

Este problema resulta ser un problema cuadrático convexo cuyo dual viene dado por:

$$\begin{aligned} \text{mín} \quad & \sum_{i=1}^{p} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{p}, \lambda_i\lambda_j y_i y_j \phi(x_i)^t \phi(x_j) \\ \text{s.a.} \quad & \sum_{i=1}^{p} y_i\lambda_i = 0 \qquad i = 1, ..., p \\ & \lambda_i \geq 0 \end{aligned}$$

Como puede apreciarse en el problema anterior, no es necesario conocer de manera explícita la función de inmersión ϕ, sino que basta disponer de un procedimiento para evaluar los productos escalares $\phi(x_i)^t\phi(x_j)$. Para ello se utilizan las *funciones núcleo o* Kernel, [2, 51, 27].

Un Kernel es una función $K : X \times X \rightarrow F$ con $X \subset \mathbb{R}^n$, tal que $K(x, z) = \phi(x)^t\phi(z)$ $x, z \in X$ donde ϕ es una transformación del espacio de entrada X en el espacio de características F.

Las funciones *Kernel* permiten el cálculo de los productos escalares $\phi(x_i)\phi(x_j)$ en el espacio de entrada $\mathbb{X}$, en lugar de en el espacio transformado F (lo que se conoce como *Kernel trick*), por lo que no se precisa conocer la función ϕ, [13, 51].

El siguiente teorema caracteriza las funciones *Kernel*:

Teorema de Mercer: Para cualquier función $K : X \times X \rightarrow \mathbb{R}$ que sea simétrica y semidefinida positiva, existe un espacio de Hilbert F y una función $\phi : X \rightarrow F$ tal que:

$$K(x, y) = \phi(x)^t\phi(y) \qquad \forall; x, y \in X$$

A continuación se presentan algunos ejemplos de funciones *Kernel*, [51, 29, 27]:

Kernel	$K(x_i, z_i)$
Polinomio de grado d	$(\langle x_i, z_i\rangle + c)^d$
Función radial Gaussiana	$exp\{-\frac{\|x_i - z_i\|^2}{2\sigma^2}\}$
Laplaciano	$exp\{\frac{\|x_i - z_i\|}{\sigma}\}$
Thin-plate spline	$\left(\frac{\|x_i - z_i\|}{\sigma}\right)^2 log_e\{\frac{\|x_i - z_i\|}{\sigma}\}$
Sigmoid	$tanh(a\langle x_i, z_i\rangle + b)$
Kernel lineal	$x_i^t z_i$

El Kernel lineal es equivalente a determinar el hiperplano de margen máximo. Así pues, en términos de la función Kernel, el problema de optimización que hay que resolver es:

$$\begin{aligned} \text{mín} \quad & \sum_{i=1}^{p} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{p}, \lambda_i \lambda_j y_i y_j K(x_i x_j) \\ \text{s.a.} \quad & \sum_{i=1}^{p} y_i \lambda_i = 0 \qquad i = 1, ..., p \\ & \lambda_i \geq 0 \end{aligned}$$

4.4.4. Aplicación en la predicción de dianas de miRNA

En [41] se presenta la herramienta MultiMiTar, basada en el clasificador SVM, para la predicción de dianas de miRNA.

Muestra de aprendizaje y selección de características

Para el desarrollo de este algoritmo, la **muestra de aprendizaje** consta, al igual que en el desarrollo de NBmiRTar, descrito en la sección dedicada al clasificador Naïve Bayes, de un conjunto de ejemplos positivos y otro de ejemplos negativos. El conjunto de ejemplos positivos está constituído por 289 dúplex miRNA-RNAm validados experimentalmente y que se sabe forman una pareja funcional. El conjunto de ejemplos negativos, por su parte, está formado por 289 dúplex validados igualmente y que se sabe que no constituyen parejas funcionales.

A parte de estos dos conjuntos, se considera un conjunto independiente de 187 ejemplos positivos y 57 ejemplos negativos, también validados experimentalmente. Este conjunto se utilizará para el proceso de selección de parámetros óptimos en el clasificador SVM, que se analiza más adelante.

Para este algoritmo, el **vector de características** de cada dúplex consta de 90 características (Figura 4.12) que se encuentran presentes en los lugares de acoplamiento entre un miRNA y su diana y que de nuevo se considera que son de gran poder predictivo, ver [41]. Por ello, se considera la estructura de un dúplex particionada en las dos regiones ya mencionadas anteriormente, *Seed* y *outSeed*.

En este estudio, un nuevo **elemento a clasificar**, que será en definitiva un dúplex miRNA-RNAm candidato a ser una pareja funcional o no, vendrá dado por su vector de características correspondientes. De nuevo las dos **clases donde clasificar** serán, por lo tanto, "**dúplex funcional**" y "**dúplex no funcional**".

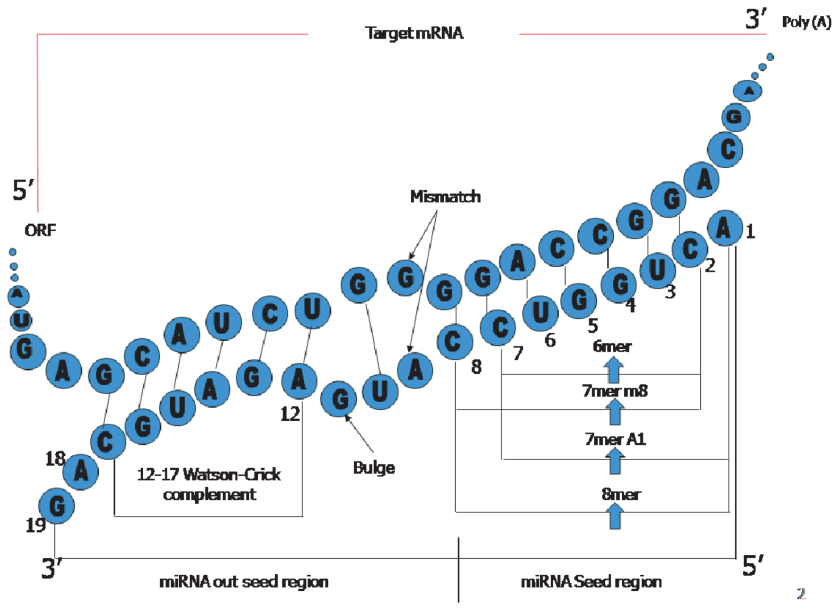

Figura 4.12: Características relevantes para el desarrollo de MultiMiTar.

Esquema y funcionamiento de MultiMiTar

MultiMiTar se basa en el clasificador SVM, por lo que se pretende generar el hiperplano de separación óptimo (ω, b) que divida los elementos de las dos clases, según la formulación de la sección anterior. Una vez se tiene este hiperplano (ω, b), un nuevo vector de entrada x se clasifica de acuerdo con:

$$f(x) = signo(\omega^T \phi(x) + b)$$

Según el signo que tome $f(x)$, -1 o 1, se clasificarán los vectores de entrada en una clase u otra. Dado que se trata un problema de elementos no separables linealmente, se utiliza la función *Kernel* radial, dada por:

$$K(x, x') = exp(-\psi||x - x'||^2)$$

Como ya sabemos, encontrar el hiperplano óptimo depende en gran medida de la selección de los parámetros C y ψ. Por ello, en el estudio se construyen diferentes modelos de clasificación para diferentes valores de C y ψ, determinando los diferentes parámetros mediante técnicas de validación cruzada, y con ellos se mide la precisión del clasificador. Los parámetros óptimos se seleccionan del modelo que presente mayor precisión a la hora de clasificar.

Resultados

Se ha comparado el funcionamiento de MultiMiTar con otros métodos predictivos, realizándose la validación de todos ellos con el conjunto independiente que se reservó al principio del experimento y realizando una comparativa en términos de las tasas de falsos positivos y falsos negativos para los diferentes algoritmos. Para ello, se utiliza al análisis ROC descrito anteriormente. La Figura 4.13 muestra un gráfico donde se compara la sensibilidad y especificidad de MultiMiTar con las de otros de los algoritmos existentes. El área del gráfico se encuentra dividida en 4 cuadrantes. La diagonal entre (0,0) y (1,1) corresponde al clasificador que produce igual tasa de falsos positivos que de falsos negativos, es decir, un clasificador completamente aleatorio. Los diferentes cuadrantes agrupan los algoritmos según:

- Cuadrante 1: Algoritmos que alcanzan una sensibilidad alta pero una baja especificidad.
- Cuadrante 2: Algoritmos que alcanzan niveles altos de especificidad y sensibilidad.
- Cuadrante 3: Algoritmos con alta especificidad pero baja sensibilidad.
- Cuadrante 4: Algoritmos que muestran baja especificidad y baja sensibilidad.

Los algoritmos que se encuentran en el cuadrante 2 son los más precisos.

En términos del Coeficiente de correlación de Matthew:

$$MCC = \frac{tp \times fn - fp \times fn}{\sqrt{(tp + fn) \times (tn + fn) \times (tp + fn) \times (tn + fp)}}$$

y de la accuracy (ACA), la comparativa anterior concluye que MultiMitar presenta $MCC = 0.58$ y $ACA = 0.8$ (Figura 4.14).

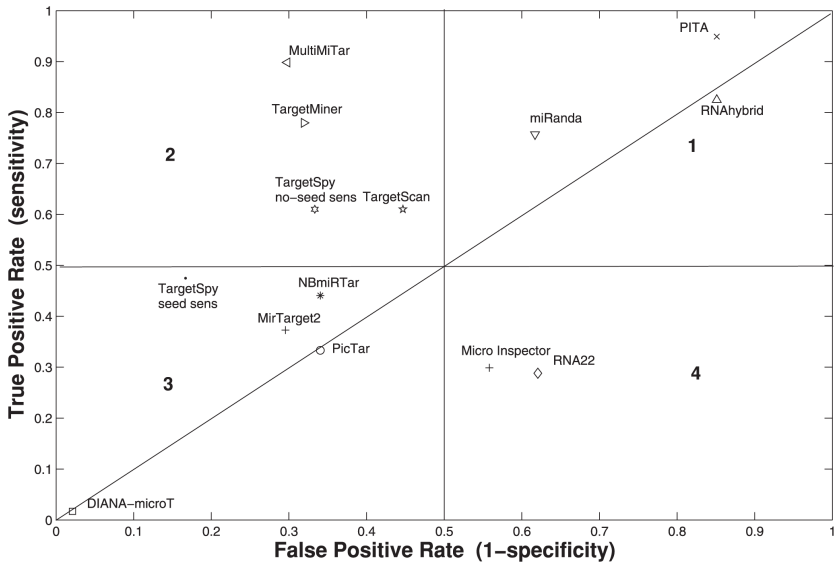

Figura 4.13: Comparativa entre algoritmos predictivos.

Method	MCC	ACA
MultiMiTar	0.583	0.800
TargetMiner	0.403	0.730
PITA	0.155	0.549
TargetScan	0.135	0.582
miRanda	0.128	0.570
NBmiRTar	0.083	0.550
MirTarget2	0.052	0.495
PicTar	−0.006	0.496
DIANA MicroT 3.0	−0.013	0.498
RNAhybrid	−0.029	0.487
MicroInspector	−0.216	0.378
RNA22	−0.269	0.321
TargetSpy no-seed sens	0.209	0.560
TargetSpy no-seed spec	0.209	0.560
TargetSpy seed sens	0.234	0.557
TargetSpy seed spec	0.234	0.557

doi:10.1371/journal.pone.0024583.t002

Figura 4.14: Comparativa en términos de MCC y ACA.

4.5. Clasificador k-NN

A continuación se introduce un algoritmo muy intuitivo de fácil implementación, basado en criterios de **proximidad entre elementos**, [33, 1].

El algoritmo de clasificación supervisada k-NN, del inglés *k-Nearest Neighbors*, [52, 22, 53, 17], realiza la clasificación de un nuevo ejemplo basándose en los ejemplos de la muestra de aprendizaje más cercanos al que se pretende clasificar; de forma más concreta, se asigna al nuevo ejemplo la clase más frecuente entre los k vecinos más próximos. A diferencia de otros clasificadores como, por ejemplo, SVM, el algoritmo k-NN no genera un modelo de clasificación explícito, recayendo todo el esfuerzo computacional en la fase de clasificación; es preciso indicar que este esfuerzo computacional es elevado, pues la clasificación de un ejemplo requiere el cálculo de tantas distancias como elementos constituyan la muestra de aprendizaje.

Dada una muestra de aprendizaje $T = \{(x_1, c_1), \ldots, (x_p, c_p)\}$, y un ejemplo a clasificar con vector de características y, el algoritmo k-NN se basa en el cálculo de $d(x_i, y) = d_i$ para $i = 1, .., p$. Estas p distancias son ordenadas de forma ascendente.

Fijado un valor del parámetro k, se seleccionan las k primeras distancias, y se considera el conjunto $T_k \subset T$ de elementos de la muestra de aprendizaje correspondientes a estas k distancias seleccionadas. Finalmente, se asigna al elemento y la clase más frecuente en T_k.

Algoritmo 1 Pseudocódigo del algoritmo k-NN

Entrada: Muestra de aprendizaje $\{(x_1, c_1), \ldots, (x_p, c_p)\}$, elemento a clasificar y.

1: **Para** $i = 1, \ldots, p$ **hacer**
2: Calcular $d_i = d(x_i, y)$.
3: **fin Para**
4: Ordenar $\{d_i, \quad i = 1, \ldots, p\}$ en orden ascendente.
5: Seleccionar los k primeros elementos de la lista ordenada.
6: Tomar $T_k \subset T$, conjunto de ejemplos correspondientes.
7: Asignar a y la clase más frecuente en T_k.

La utilización más sencilla de este algoritmo corresponde al caso $k = 1$, es decir, al algoritmo 1-*NN*. El criterio de clasificación en este caso consiste simplemente en asignar al nuevo ejemplo la clase del elemento más cercano de la muestra de aprendizaje (Figura 4.15).

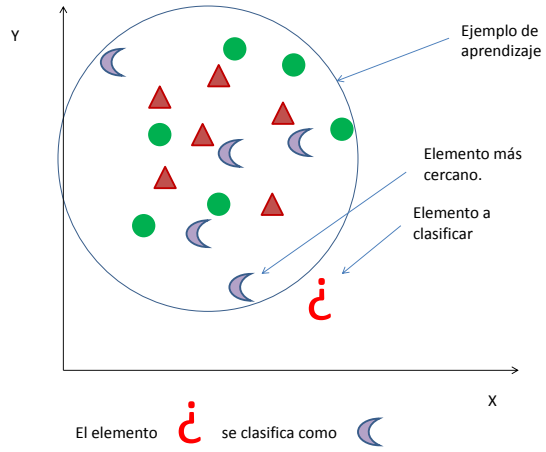

Figura 4.15: Algoritmo 1-NN.

Para $k > 1$ el procedimiento es igual de intuitivo, como puede comprobarse en la Figura 4.16 para el caso $k = 3$.

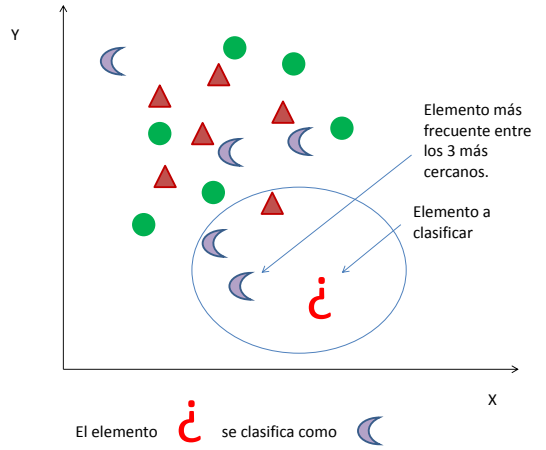

Figura 4.16: Algoritmo 3-NN.

Los dos elementos claves del algoritmo k-NN son el parámetro k y la distancia utilizada para medir la proximidad entre ejemplos. Usualmente se considera la **distancia euclídea** que, para dos vectores $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ e $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$, viene dada por:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Alternativamente, pueden utilizarse otras distancias más adecuadas para el tipo de datos utilizados. Algunas distancias utilizadas habitualmente son las siguientes:

- **Distancia de Manhattan o distancia Taxi**:

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

- **Distancia de Chebychev**:

$$d(x,y) = \max_{i=1,\ldots,n} |x_i - y_i|$$

- **Distancia discreta**:

$$d(x,y) = \begin{cases} 1 & \text{si } x \neq y \\ 0 & \text{si} x = x \end{cases}$$

- **Distancia de Hamming**, para datos categóricos:

  $d(x,y) = |\{i \ : \ x_i \neq y_j\}|$ = número de componentes distintas entre x e y.

- **Distancia L_p (ponderada)**:

$$d(x,y) = \left(\sum_{i=1}^{n} \omega_i |x_i - y_i|^p \right)^{1/p}$$

- **Distancia de Malahanobis**:

$$d(x,y) = \sqrt{(x-y)^T S^{-1} (x-y)}$$

  Donde $S \in M_{n \times n}$

La elección del valor de k es el otro aspecto esencial del algoritmo k-NN. El ejemplo más claro de inestabilidad se da para k=1, pues en el algoritmo 1-NN sólo tiene en cuenta el elemento más cercano. La elección del k para evitar problemas de inestabilidad depende de la distribución de los datos en cada problema y no hay ninguna regla específica para determinarlo más que probando diferentes valores.

4.5.1. Variantes del algoritmo k-NN

Se pueden considerar algunas variantes del algoritmo k-NN como pueden ser:

- **k-NN con rechazo**: En esta variante el nuevo ejemplo se queda sin clasificar si no se cumple una cierta condición, establecida previamente, que garantice de alguna manera que la clase en la que se clasifica es la correcta. Por ejemplo, puede establecerse que la clasificación solo se lleve a cabo si la frecuencia de la clase mayoritaria en los k ejemplos más cercanos supera un cierto umbral U_1 o si la diferencia entre la frecuencia de la clase mayoritaria y la siguiente supera un cierto umbral U_2.

- **k-NN con distancia media**: Para esta variante es preciso calcular la distancia media a los elementos de cada clase en el conjunto T_k. El nuevo elemento se clasifica en la clase con menor distancia media. En la Figura 4.17 se aprecia que a pesar de que cuatro de los seis elementos más cercanos al elemento a clasificar pertenecen a la clase $\times$, el elemento se clasifica como **O**, ya que la distancia media del nuevo elemento a los elementos **O** es menor que la distancia al resto de elementos.

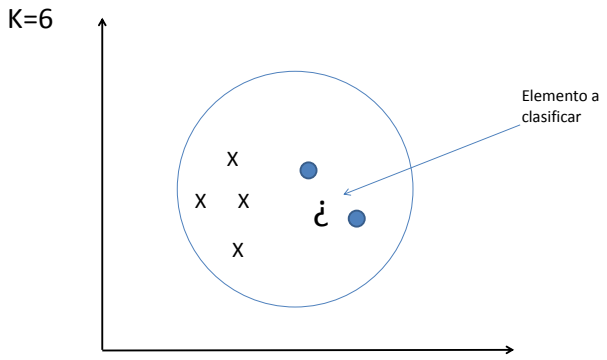

Figura 4.17: K-NN con distancia media (m=2).

- **k-NN con distancia mínima**: En el k-NN con distancia mínima se comienza seleccionando un elemento por clase, normalmente el ejemplo más cercano al centro de gravedad de todos los elementos de dicha clase. A continuación se asigna el nuevo elemento a clasificar a la clase cuyo representante esté más cercano. Este procedimiento puede verse como un 1-NN aplicado a un conjunto de m ejemplos (uno por cada clase). El coste computacional de esta variante es claramente menor

que en el caso del k-NN genérico, pues se trabaja con menos elementos. La efectividad de este algoritmo depende de la homogeneidad de las clases, funcionando mejor cuanto más homogéneas sean éstas.

- **k-NN con ponderación de ejemplos**: A cada uno de los ejemplos en T_k se le asigna un peso que tiene en cuenta la distancia que lo separa del nuevo elemento y que se pretende clasificar. Por ejemplo, este peso puede tomarse igual al inverso de la distancia que los separa o del cuadrado de ésta:

$$x_i \in T_k \rightarrow \omega_i = \frac{1}{d(x_i, y)} \quad \text{ó} \quad \omega_i = \frac{1}{d^2(x_i, y)}$$

  Para cada clase se calcula su peso total como la suma de los pesos de los ejemplos en T_k que pertenecen a esa clase y se asigna el nuevo elemento a clasificar a la clase que presente mayor peso (Figura 4.18).

Clase	Distancia	Peso
A	5	0.040
B	4	0.063
A	3	0.111
A	4	0.063
B	2	0.250

Peso (A) = 0.2136

La nueva instancia se clasifica en B

Peso (B) = 0.3125

Figura 4.18: K-NN con ponderación de ejemplos.

- **k-NN con ponderación de variables**:

  En esta variante se asigna un peso a cada variable, que está en función de la relación que exista entre la variable y el atributo de clase.

  Con ello se trata de controlar la influencia de variables poco relevantes para la clasificación. Cada peso se determina a partir de la cantidad de información mutua entre la variable y el atributo de clase.

Existen varias aproximaciones con las que se pretende paliar la fuerte dependencia del coste computacional del algoritmo k-NN con respecto del número de elementos de

la muestra de aprendizaje. Una posible solución es reemplazar la muestra de aprendizaje por otra más pequeña, cuyos elementos se denominan **prototipos**. Dos enfoques diferentes para seleccionar la nueva muestra consisten en: (1) Suprimir instancias correspondientes a observaciones "extrañas" (edición de prototipos) y (2) Seleccionar un subconjunto de instancias de la muestra de aprendizaje original en el que la regla de clasificación tenga un comportamiento similar (condensación de prototipos).

- **Edición de Wilson**

  La idea en la edición propuesta por Wilson es el someter a prueba a cada uno de los elementos de la muestra de aprendizaje. Para ello, para cada ejemplo se compara su clase verdadera con la que propone un clasificador k-NN obtenido con todos los ejemplos excepto él mismo. En el caso de que ambas clases no coincidan, el ejemplo es eliminado. Este método presenta una cierta analogía con la técnica de validación *leave-one-out*, [58].

Algoritmo 2 Pseudocódigo Edición de Wilson

Entrada: Muestra de aprendizaje $T = \{(x_1, c_1), \ldots, (x_p, x_p)\}$.

1: **Para** $i = 1, \ldots, p$ **hacer**
2: Aplicar k-NN a $T - x_i$ para clasificar x_i. Sea c^* la clase pronosticada.
3: **si** $c_i \neq c^*$ **entonces**
4: Marcar (x_i, c_i) para su eliminación.
5: Eliminar de T los ejemplos marcados.
6: **fin si**
7: **fin Para**

Salida: T' muestra de prototipos.

- **Condensación de Hart**

  Un conjunto $T_c \subset T$ se dice consistente respecto a T si la regla de clasificación k-NN basada en T_c clasifica correctamente a todos los elementos de T, [26]. La condensación de Hart intenta determinar un subconjunto consistente dentro de la muestra de aprendizaje según el siguiente algoritmo (Figura 4.19).

Algoritmo 3 Pseudocódigo Condensación de Hart

Entrada: Muestra de aprendizaje $T = \{(x_1, c_1), \ldots, (x_p, c_p)\}$

Para $i = 1, \ldots, p$ **hacer**
  Considerar $T_c = (x_i, c_i)$, $T_{aux} = T - T_c$
  Para $(x_j, c_j) \in T_{aux}$ $j \neq i, j = 1, \ldots, p$ **hacer**
    Clasificar x_i usando 1-NN a partir de T_c. Sea c^* la clase pronosticada.
    si $c_i \neq c^*$ **entonces**
      $T_{aux} = T_{aux} - (x_i, c_i)$
      $T_c = T_c \cup (x_i, c_i)$
    fin si
  fin Para
fin Para

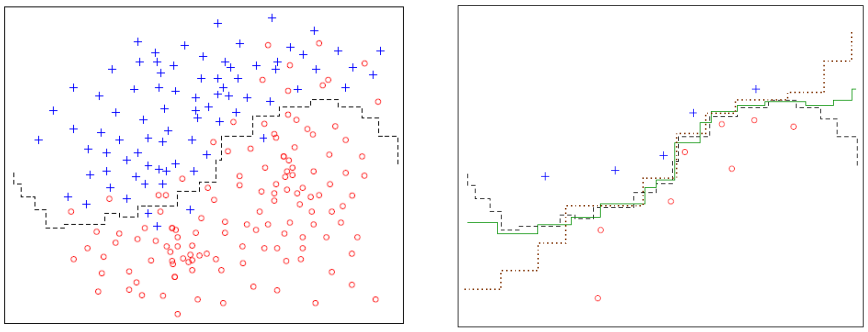

Figura 4.19: Idea sobre el tratamiento con datos condensados.

4.5.2. Aplicación en el tratamiento de datos de miRNA

Un aspecto de interés en relación con la función y expresión de los miRNAs es su uso como indicadores de presencia de tumores.

En [31] se presenta un estudio donde se analizan simultáneamente todos los miRNAs conocidos en humanos así como sus perfiles de expresión en distintos tipos de cáncer. El resultado de este trabajo revela que es posible asociar perfiles de expresión de miRNA concretos a tipos de cáncer determinados. Los autores encuentran que, en general, en los tejidos tumorales hay una menor expresión de los miRNAs en comparación con los tejidos sanos. Este hecho plantea la idea de que el papel principal de los miRNAs puede ser el de facilitar la diferenciación celular inhibiendo así la proliferación celular característica del cáncer.

Existen métodos para medir la magnitud de la expresión de los miRNA en una célula, quedando identificadas estas magnitudes por una serie de valores concretos (Figura 4.20).

Description	Sample 1	Sample 2
hsa-miR-124a:UUAAGGCACGCGGUGAAUGCCA:bead_101-A	7.4204	6.931
hsa-miR-125b:UCCCUGAGACCCUAACUUGUGA:bead_102-A	10.8391	11.7231
hsa-miR-7:UGGAAGACUAGUGAUUUUGUU:bead_103-A	6.64631	6.78163
hsa-let-7g:UGAGGUAGUAGUUUGUACAGU:bead_104-A	9.86267	10.4861
hsa-miR-16:UAGCAGCACGUAAAUAUUGGCG:bead_105-A	10.6879	11.5479
hsa-miR-99a:AACCCGUAGAUCCGAUCUUGUG:bead_107-A	8.39361	8.88749
hsa-miR-92:UAUUGCACUUGUCCCGGCCUGU:bead_108-A	8.63981	9.06636

Figura 4.20: Valores indicadores de la expresión de miRNA.

Trabajando sobre la base de datos generada en [31], formada por valores de expresión de 217 miRNAs encontrados en 186 tejidos, en [35] se intenta incrementar el valor informativo de los perfiles de miRNA incorporando, de modo previo al uso de clasificadores, métodos de selección de atributos o características para seleccionar, del conjunto de miRNAs de la base de datos, los asociados a los estados (o clases, si hablamos en términos de un problema de clasificación) "normal" / "cáncer", que puede presentar un tejido.

Métodos de selección de características

- **Métodos basados en similitud**

  En estos métodos, a cada miRNA se le asocia un vector cuyas componentes corresponden a los valores de expresión del miRNA en las distintas muestras (Figura 4.21).

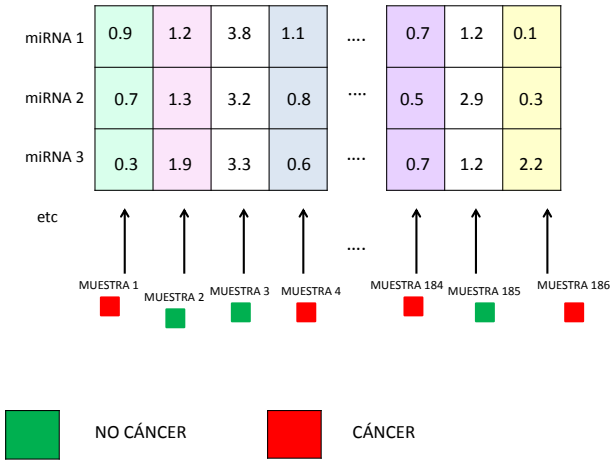

Figura 4.21: Vectores de expresión de miRNA para cada muestra.

También se consideran los denominados **vectores ideales**. El vector ideal positivo es aquel cuya i-ésima componente vale 1 si corresponde a un tejido con cáncer y 0 en caso contrario. El vector ideal negativo se construye al revés: su componente i-ésima vale 0 si la muestra corresponde a un tejido con cáncer y 1 en caso contrario (Figura 4.22).

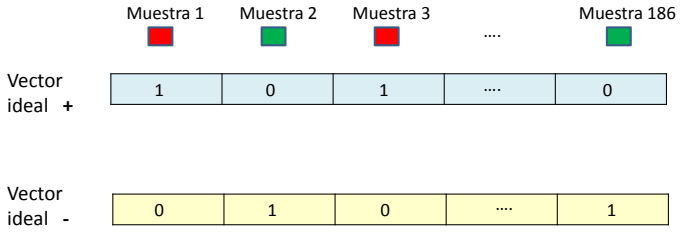

Figura 4.22: Vectores ideales para clasificación

El procedimiento que siguen estos métodos consiste en ir comparando los vectores de expresión de cada miRNA con estos vectores ideales. Previamente, los valores de expresión de miRNA son normalizados de la siguiente manera: para cada

miRNA se determinan los valores máximos y mínimos en el conjunto de muestras, $g_{\text{máx}}$ y $g_{\text{mín}}$; cada valor de expresión del miRNA considerado, g, es normalizado de acuerdo con $\dfrac{g - g_{\text{mín}}}{g_{\text{máx}} - g_{\text{mín}}}$, por lo que todos los valores pasan a estar contenidos en el intervalo [0,1]. Una vez normalizados todos los miRNAs, se ordenan de acuerdo con la similitud que presentan con los vectores ideales, obteniéndose dos grupos, según que se encuentran más próximos al vector ideal positivo o al negativo. Finalmente, se forma un conjunto de miRNAs donde la mitad provienen del conjunto de miRNAs más cercanos al vector ideal positivo y la otra mitad corresponden a miRNAs con mayor similitud al vector ideal negativo.

Se usan cuatro medidas de similitud entre vectores: la inversa de la distancia euclídea (IE), similitud coseno, correlación de Pearson y correlación de Spearman. En este proceso, el número de miRNAs mejor clasificados que se seleccionaron fue 25, [14].

- **Similitud coseno (CC)**: Es una medida de la similitud existente entre dos vectores en un espacio que posee un producto interior con el que se evalúa el coseno del ángulo comprendido entre ellos. Un valor de esta función igual a 1 indica vectores con un ángulo entre ellos igual a 0. Un valor de -1 indica vectores con sentidos opuestos. El valor de esta función de similitud oscila, por lo tanto, entre -1 y 1.

$$f_{cos}(x_1, x_2) = ||x_1|| \cdot ||x_2|| \cdot \cos(\widehat{x_1, x_2})$$

- **Correlación de Pearson (CP)**: Es una medida de la relación lineal entre dos variables aleatorias cuantitativas

$$\rho_{X,Y} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

Donde σ_{XY} es la covarianza de (X, Y), σ_X es la desviación típica de X y σ_Y es la desviación típica de Y. El coeficiente de correlación de Pearson oscila entre -1 y 1. Si $\rho = 1$ existe una dependencia lineal total directa entre las dos variables: cuando una aumenta, lo hace en proporción constante la otra. Si $\rho = -1$ existe una dependencia lineal total inversa, es decir, cuando una aumenta, la otra disminuye en proporción constante.

- **Correlación de Spearman (CS)**: Es una medida de la relación o interdependencia entre dos variables aleatorias X e Y. Para calcular $\hat{\rho}$, los datos son ordenados y reemplazados por su respectivo orden.

$$\hat{\rho} = 1 - \frac{6 \sum D^2}{N(N^2 - 1)}$$

donde D representa la diferencia entre los correspondientes estadísticos de orden de X e Y (rangos) y N representa el número de parejas. La interpretación del coeficiente de correlación de Spearman es la misma que la del coeficiente de correlación de Pearson.

- **Relación señal-ruido (SN)**: Cuando se pretende hacer referencia a la relación entre la magnitud de la media y la variabilidad de la variable, se utiliza el coeficiente de variación. Su fórmula expresa la desviación estándar como fracción de la media aritmética,

$$C_v = \frac{\sigma}{|\mu|}$$

La relación señal-ruido puede interpretarse como la inversa del coeficiente de variación. Si calculamos la media μ y la desviación típica σ de la distribución de los valores de expresión de miRNAs dentro de cada clase, el coeficiente de relación señal-ruido del valor de expresión de un miRNA g_i se define como sigue:

$$SN(g_i) = \frac{|\mu_{NORMAL}(g_i) - \mu_{CANCER}(g_i)|}{\sigma_{NORMAL}(g_i) + \sigma_{CANCER}(g_i)}$$

- **Información mutua (MI)**: la información mutua o transinformación de dos variables aleatorias es una cantidad que mide la dependencia mutua de las dos variables, es decir, mide la reducción en la incertidumbre (entropía) de una variable aleatoria, $X = (x_i)$, debida al conocimiento del valor de otra variable aleatoria $Y = (y_j)$.

$$MI(x_i, y_j) = log \frac{P(x_i|y_j)}{P(x_i)} \qquad i = 1, \ldots, n \quad j = 1, \ldots, m$$

En el estudio, se calcula la información mutua entre cada miRNA y la variable de clase ("normal" o "cáncer"), pero categorizando en dos grupos los valores de miRNA, según sean mayores o menores que la media de todos los miRNA en el conjunto de aprendizaje. Si TN es el número total de ejemplos de la muestra de aprendizaje, se tiene:

$$\begin{aligned} MI(g_i) &= MI(g_i \geq \hat{g}_i, t_i = \text{“normal”}) + \\ &\quad MI(g_i \geq \hat{g}_i, t_i = \text{“cáncer”}) + \\ &\quad MI(g_i < \hat{g}_i, t_i = \text{“normal”}) + \\ &\quad MI(g_i < \hat{g}_i, t_i = \text{“cáncer”}) \end{aligned}$$

siendo

$$\hat{g}_i = \frac{1}{TN} \sum_{j=1}^{TN} g_{j,i}$$

$$MI(g_i > \hat{g}_i, t_i = \text{“normal”}) =$$

$$P(g_i > \hat{g}_i, t_i = \text{“normal”}) log_{10} \frac{P(g_i > \hat{g}_i, t_i = \text{“normal”})}{P(g_i > \hat{g}_i) \times P(t_i = \text{“normal”})}$$

Donde $g_{j,i}$ hace referencia al valor de expresión del miRNA i en la muestra j.

Entrenamiento de k-NN

Tras este paso previo de selección de atributos, se emplean diferentes métodos para clasificar una muestra de tejidos en las dos clases posibles, “Cáncer” o “Normal”. Entre todos los clasificadores que se utilizan, se entrenan cuatro variantes del algoritmo k-NN, con $k = 3$, usando 4 medidas de similitud distintas: la inversa de la distancia euclídea (**KNNE**), correlación de Pearson (**KNNP**), coeficiente coseno (**KNNC**) y correlación de Spearman (**KNNS**), [35].

Resultados

En [31] se han considerado las diferentes combinaciones posibles entre clasificadores k-NN y algoritmos de selección de atributos (Figura 4.23).

Los resultados experimentales concluyeron que un clasificador KNNS junto con un método de selección de atributos basado en similitud tomando como medida la similitud coseno, proporcionan un *accuracy* máxima del 95 %.

	IE	CP	CC	CS	MI	SN
KNNE	92.7	92.4	91.7	90.0	93.0	90.8
KNNP	93.3	86.4	94.1	87.5	91.7	93.2
KNNP	92.2	85.9	93.2	87.7	90.6	90.9
KNNC	93.0	85.3	95.0	87.9	90.9	91.2

Figura 4.23: Accuracy según se combinan diferentes k-NN con métodos de selección de características. Las columnas corresponden a los diferentes métodos de selección de características basados en la inversa de la distancia euclídea (IE), la correlación de Pearson (CP), la Similitud Coseno (CC), la Correlación de Spearman (SP), la información mutua (MI) y la relación señal-ruido (SN).

4.6. Redes neuronales

Las redes de neuronas artificiales constituyen una técnica de aprendizaje automático inspirada en el comportamiento del cerebro humano.

La neurona es el elemento básico del sistema nervioso humano: poseemos más de 10^{11} neuronas compartiendo entre ellas unas 10^{15} conexiones. Lo que hace únicas a las células del sistema nervioso, en comparación con las de otros sistemas del cuerpo humano, es su capacidad de recepción, proceso y transmisión de señales electroquímicas a través de ellas mismas y de sus conexiones. La mayoría de las neuronas consisten en un cuerpo celular unido a un axón y a varias dendritas (Figura 4.24). Funcionalmente hablando, las señales llegan a las dendritas procedentes de los axones de otras células a través de puntos de conexión llamados sinapsis o conectores sinápticos. Una vez allí, las señales recibidas pasan al cuerpo celular donde son combinadas con otras señales (provenientes de otras dendritas de la célula). Si, en un cierto período de tiempo, el resultado de esta combinación excede un cierto valor límite, la célula se activa, lo que se traduce en un impulso de salida que recorre el axón y se propaga por las sinapsis correspondientes. De esta manera, cada neurona recibe, vía sus dendritas, impulsos de cientos de otras neuronas y envía su propio pulso, resultado de éstos, a cientos de otras más. Es esta complejidad de conexión, más que el trabajo realizado por cada neurona, la que posibilita la realización de tareas como las que habitualmente son adscritas a los seres inteligentes.

Debido a su constitución y a sus fundamentos, las redes neuronales artificiales pre-

sentan un gran número de características semejantes a las del cerebro. Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se esté aplicando en múltiples áreas. Entre sus ventajas se incluyen:

- Aprendizaje adaptativo. Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.
- Auto-organización. Una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.
- Tolerancia a fallos. La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.
- Operación en tiempo real. Los cómputos neuronales pueden ser realizados en paralelo; para esto se diseñan y fabrican máquinas con hardware especial para obtener esta capacidad.

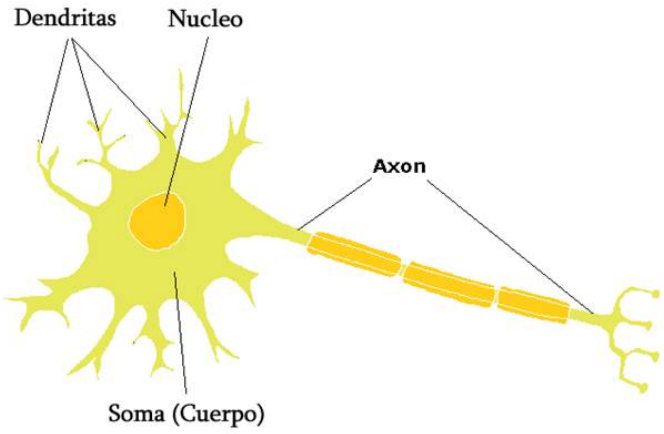

Figura 4.24: Dibujo esquemático de una neurona real.

4.6.1. Elementos básicos que componen una red neuronal

Una neurona artificial es una estructura procesadora de información, distribuida y paralela, que tiene la forma de un grafo dirigido, como se aprecia en la Figura 4.25. Recuérdese que un grafo dirigido es una estructura compuesta por un conjunto de

puntos (llamados nodos) y un conjunto de segmentos de línea dirigidos (llamados arcos o aristas) que los conectan.

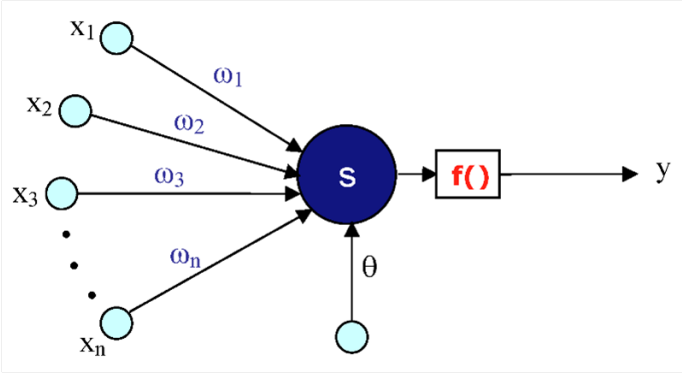

Figura 4.25: Estructura de una neurona artificial.

La estructura de una neurona artificial consiste en:

- Un conjunto de **nodos de entrada** x_i**,** $i = 1, ..., m$.
- Los arcos, denominados **conexiones** (caminos propagadores de la señal, unidireccionales e instantáneos), con sus respectivos **pesos sinápticos** ω_i**,** $i = 1, ..., m$ donde se almacena la información que determinará la funcionalidad de la red. Estos pesos se modificarán en el **entrenamiento de la red neuronal**.
- Una **función de propagación** ***S*** que combina las entradas y los pesos sinápticos. Por ejemplo, $S(x_1, x_2, x_3, ..., x_m, \omega_1, \omega_2, ..., \omega_m) = \sum_{i=1}^{m} x_i \omega_i$
- Una **función de activación** ***f***, función de S y de una constante Θ denominada **umbral o sesgo**, que proporciona la salida y de la neurona. El número de salidas de una neurona puede ser cualquiera, pero todas portan la misma información y a su vez se conectan con las entradas de otra neurona.

Suelen usarse diferentes tipos de funciones de activación (Figura 4.26).

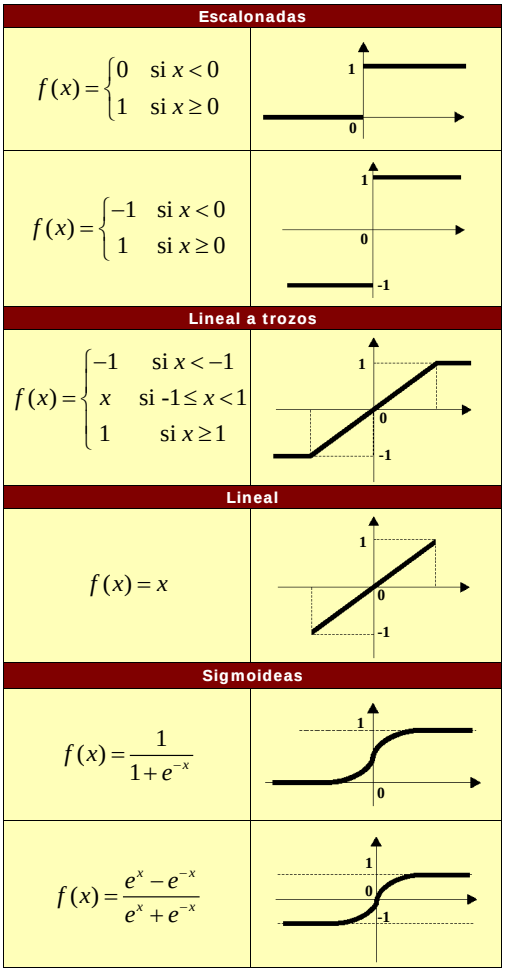

Figura 4.26: Diferentes funciones de activación.

La topología o **arquitectura** de las redes neuronales hace referencia a la organización y disposición de las neuronas en la red. La arquitectura de una red neuronal depende de cuatro parámetros principales: (1) El número de capas del sistema; (2) el número de neuronas por capa; (3) el grado de conectividad entre las neuronas; (4) el tipo de conexiones neuronales, [20].

Los tipos de capas que puede presentar una red neuronal son (Figura 4.27):

- Capa de entrada: formada por las neuronas que reciben la información destinada a la red.

- Capa de salida: formada por las neuronas encargadas de transmitir la salida de respuesta de la red.
- Capas ocultas: en estas capas se agrupan las neuronas restantes.

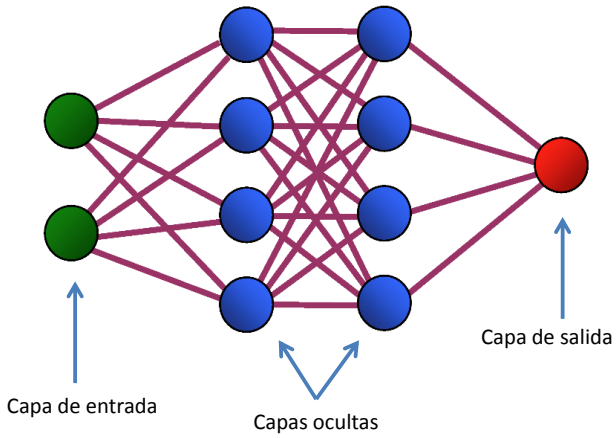

Figura 4.27: Tipos de capas en una red neuronal.

Según el número de capas que posea una red, ésta puede ser una **red monocapa** o una **red multicapa** y a su vez, según sea el flujo de datos, éstas pueden ser **unidireccionales** o **realimentadas** (Figura 4.28).

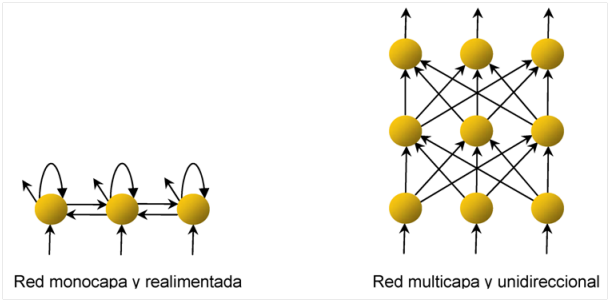

Figura 4.28: Ejemplos de redes neuronales.

4.6.2. Aprendizaje de una red neuronal

Antes de comenzar el proceso de entrenamiento se debe determinar un estado inicial, es decir, escoger un conjunto inicial de pesos para las diversas conexiones entre las

neuronas de la red neuronal. Esto puede realizarse de diversas formas; por ejemplo, puede asignarse un peso aleatorio a cada conexión, encontrándose los mismos dentro de un cierto intervalo, generalmente de la forma $[-n, n]$, donde n es un número natural positivo, [21].

Existen diversos esquemas de aprendizaje para la red neuronal, es decir, diferentes técnicas que permiten entrenar la red a partir de un conjunto de aprendizaje, determinando los parámetros libres:

- Aprendizaje por corrección de error. Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida. Un ejemplo de este tipo de algoritmos lo constituye la regla de aprendizaje del Perceptrón.
- Aprendizaje por refuerzo. Se trata de un aprendizaje supervisado, más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada.
- Aprendizaje estocástico. Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.
- Aprendizaje no supervisado. Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.

Antes de comenzar a entrenar la red se debe establecer la condición de parada del proceso de entrenamiento. Algunas de las condiciones utilizadas habitualmente son:

- Se ha alcanzado una cuota de error que consideramos suficientemente pequeña.
- Se ha alcanzado un número máximo de iteraciones definido como límite para el entrenamiento.
- Se ha obtenido un error cero.
- Se ha llegado a un punto de saturación en el que por más que entrenemos ya no se consigue reducir el error.

4.6.3. Tipos de redes neuronales

Dependiendo del modelo de neurona concreto que se utilice, de la arquitectura o topología de conexión, y del algoritmo de aprendizaje, surgirán distintos modelos de redes neuronales (Figura 4.29).

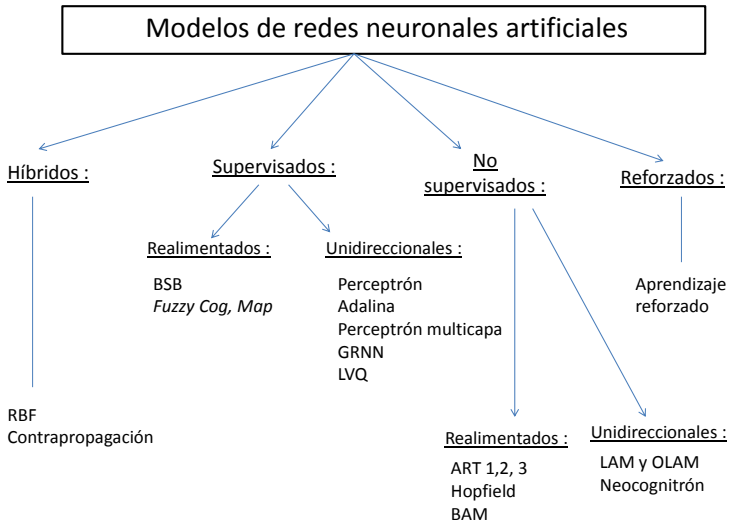

Figura 4.29: Clasificación de las redes neuronales según el tipo de aprendizaje y arquitectura.

El perceptrón simple

Una de las características más significativas de las redes neuronales es su capacidad para aprender a partir de alguna fuente de información interactuando con su entorno. Ronsenblatt [47] desarrolló un modelo simple de neurona basado en el modelo de McCulloch y Pitts [39] y en una regla de aprendizaje basada en la corrección del error. A este modelo le llamó Perceptrón. Una de las características que más interés despertó de este modelo fue su capacidad de aprender a reconocer patrones. El Perceptrón está constituido por un conjunto de neuronas de entrada que reciben los patrones de entrada a reconocer o clasificar y una neurona de salida que se ocupa de clasificar a los patrones de entrada en dos clases, según que la salida de la misma sea 1 (activada) o 0 (desactivada).

El perceptrón simple (Figura 4.30) es un sistema capaz de realizar tareas de clasificación de forma automática mediante el **Algoritmo de Rosenblatt**. A partir de la muestra de aprendizaje, el sistema genera la ecuación de un hiperplano discriminante, cuya ecuación viene dada por:

$$\omega_1 x_1 + \omega_2 x_2 + \Theta = 0 \tag{4.15}$$

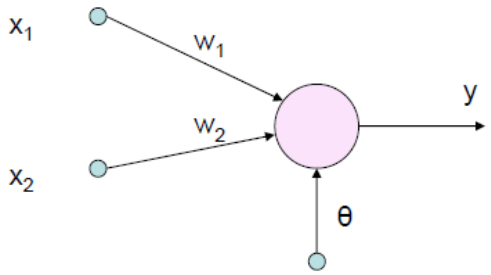

Figura 4.30: Ejemplo de perceptrón simple.

Podemos calcular el valor de la salida del siguiente modo:

$$y = \begin{cases} 1 & \text{si } \omega_1 x_1 + \omega_2 x_2 + \Theta > 0 \\ -1 & \text{si } \omega_1 x_1 + \omega_2 x_2 + \Theta \leq 0 \end{cases}$$

En el caso de que la salida sea +1, la entrada pertenecerá a una clase, situándose el punto (x_1, x_2) en uno de los semiplanos determinado por (4.15). Si la salida es -1, la entrada pertenecerá a la clase contraria, situándose el punto (x_1, x_2) en el semiplano contrario.

El perceptrón simple presenta una serie de limitaciones, siendo la más importante su incapacidad para clasificar correctamente conjuntos que no son linealmente separables.

Algoritmo de Rosenblatt

El algoritmo de Rosenblatt pertenece al conjunto de los algoritmos basados en la correción de errores.

Supongamos un perceptrón simple con n neuronas de entrada y m neuronas de salida, en el que tanto las variables de entrada como las de salida solo pueden tomar los valores $\{-1, 1\}$. Dada un conjunto de aprendizaje de tamaño N, sean x_i^r los valores de entrada y t_j^r los valores de salida reales para el elemento r-ésimo de dicho conjunto de aprendizaje, $r = 1, \ldots, N$, $i = 1, \ldots, n$, $j = 1, \ldots, m$. Representaremos por y_j^r las salidas proporcionadas por la red para cada elemento del conjunto de aprendizaje. La idea del algoritmo de Rosenblatt es ir ajustando los pesos de cada nodo de manera que y^r vaya siendo cada vez más parecido a t^r.

La actualización (aumento o disminución) de cada peso en cada iteración viene dada por:

$$w_{ij} = w_{ij} + \Delta_{ij}^{r}$$

siendo $\Delta_{ij}^{r} = \varepsilon(t_j^r - y_j^r)x_i$

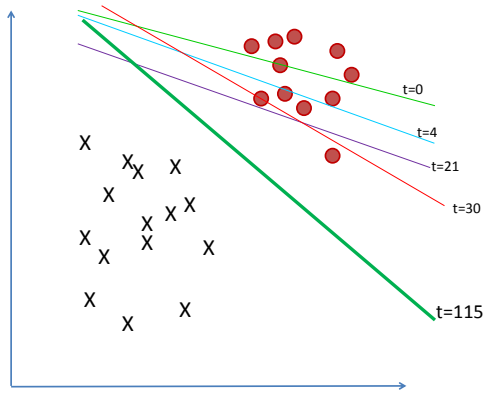

Figura 4.31: Evolución de las regiones de decisión según el algoritmo de Rosenblatt.

El parámetro ε, $0 < \epsilon \leq 1$, se denomina *coeficiente o tasa de aprendizaje*. Un valor pequeño de ε implica un aprendizaje lento, mientras que un valor alto puede conducir a oscilaciones excesivas de los pesos, no aconsejables en el proceso de entrenamiento.

Si los ejemplos de la muestra de aprendizaje son separables linealmente, entonces el algoritmo de Rosenblatt converge en un número finito de iteraciones para cualquier elección inicial de los pesos, mientras que si se trata de un conjunto de ejemplos no separable linealmente el proceso oscilará sin llegar a converger.

El perceptrón simple es capaz de reproducir las puertas lógicas OR y AND, pero no así la puerta XOR, ya que no es linealmente separable (Figura 4.32).

El perceptrón multicapa

Es una de las arquitecturas más utilizadas para resolver problemas reales, ya que un perceptrón multicapa puede aproximar relaciones no lineales entre los datos de entrada y salida. Se evalúa un conjunto de datos de entradas y se obtienen valores reales o vectores con valores reales. Se diferencia del perceptrón simple en que tiene una o varias capas ocultas, como puede observarse en la Figura 4.33.

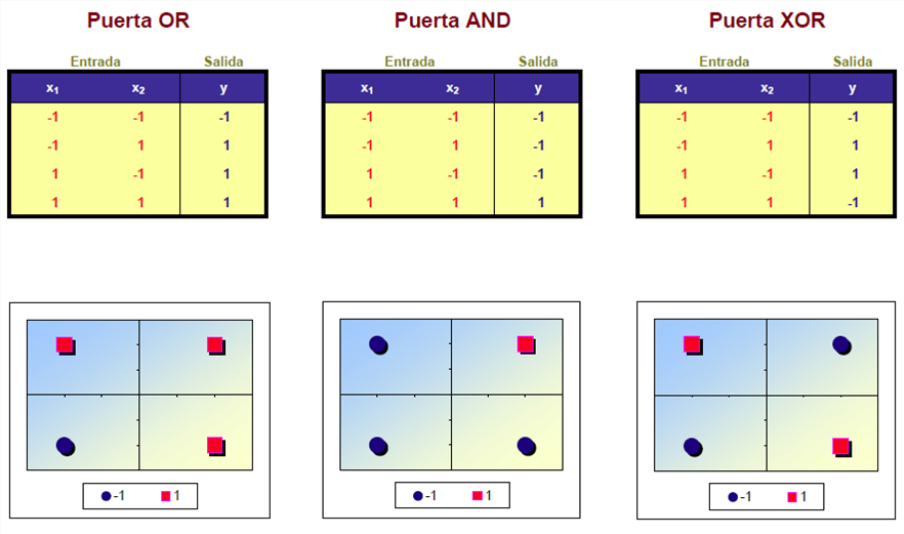

Figura 4.32: Operadores lógicos OR, AND y XOR.

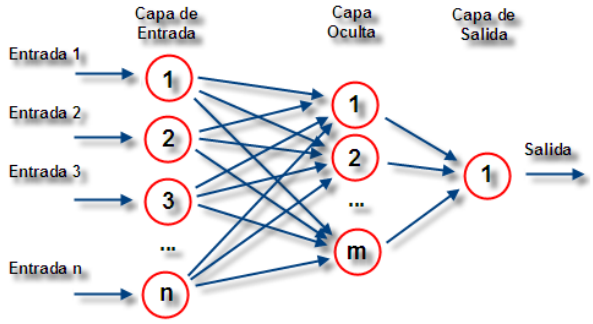

Figura 4.33: Ejemplo de perceptrón multicapa.

Este modelo se compone de:

- Capa de entrada: sólo se encarga de recibir las señales de entrada y propagarla a la siguiente capa.
- Capa de salida: proporciona al exterior la respuesta de la red para cada patrón de entrada.
- Capas ocultas: realizan un procesamiento no lineal de los datos de entrada.

La propagación de los patrones de entrada en el perceptrón multicapa define una relación entre las variables de entrada y variables de salida de la red. Esta relación se

obtiene propagando hacia delante los valores de entrada. Cada neurona de la red procesa la información recibida por sus entradas y produce una respuesta o activación que se propaga, a través de las conexiones correspondientes, a las neuronas de la siguiente capa.

El algoritmo *Backpropagation* (retropropagación)

Este método de entrenamiento se conoce también como regla Delta generalizada, y trata de hacer mínimo el error cuadrático medio cometido sobre la muestra de aprendizaje. Para su descripción, utilizaremos la siguiente notación en el caso de un perceptrón con una capa oculta.

- Capa de entrada: n neuronas
- Capa oculta: o neuronas
- Capa de salida: m neuronas
- x_i entradas, y_j salidas capa oculta, z_k salidas última capa
- ω_{ij} pesos de las neuronas ocultas, Θ_k umbrales de las neuronas ocultas
- $\hat{\omega}_{ij}$ pesos de las neuronas de salida, $\hat{\Theta}_k$ umbrales de las neuronas de salida.

En el caso lineal para las salidas de la red, z_k viene dado por,

$$z_k = \sum_{j=1}^{o} \hat{\omega}_{jk} y_j - \hat{\Theta}_k = \sum_{j=1}^{o} \hat{\omega}_{jk} f(\sum_{i=1}^{n} \omega_{ij} x_i - \Theta_j) - \hat{\Theta}_k \qquad k = 1, \ldots, m$$

Usualmente suele considerarse f una función de tipo sigmoide en la capa oculta y en la de salida.

Para el elemento r-ésimo del conjunto de aprendizaje, $r = 1 \ldots, N$, el erro cuadrático medio cometido es:

$$E_r = \frac{1}{2} \sum_{k=1}^{m} (t_k^r - z_k^r)^2 = \frac{1}{2} ||t^r - z^r||^2$$

El error cuadrático medio cometido sobre la muestra de aprendizaje es, por tanto:

$$E = \sum_{r=1}^{N} E_r = \frac{1}{2}\sum_{r=1}^{N}\sum_{k=1}^{m}(t_k^r - z_k^r)^2 = \frac{1}{2}\sum_{r=1}^{N}||t^r - z^r||^2$$

Este algoritmo minimiza E siguiendo el método del gradiente y, por lo tanto, requiere que la función f sea diferenciable. Los pesos se irán moviendo en la dirección y sentido de máximo decrecimiento según,

$$\lambda := \lambda - \varepsilon\Delta E(\lambda) \qquad \lambda = (\omega, \hat{\omega}, \Theta, \hat{\Theta})$$
$$E(\lambda) = \sum_{r=1}^{m} E_r(\lambda) \rightarrow \Delta E(\lambda) = \sum_{r=1}^{N} \Delta E_r(\lambda)$$

Realizando los cálculos oportunos, se obtiene:

$$\begin{aligned}
E_r &= \frac{1}{2}\sum_{k=1}^{m}(t_k^r - z_k^r)^2 = \frac{1}{2}\sum_{k=1}^{m}\Big(t_k^r - \Big(\sum_{j=1}^{o}\hat{\omega}_{ij}y_j^r - \hat{\Theta}_k\Big)\Big)^2 = \\
&= \frac{1}{2}\sum_{k=1}^{m}\Big(t_k^r - \sum_{j=1}^{o}\hat{\omega}_{ij}f\Big(\sum_{i=1}^{n}\omega_{ij}x_i^r - \Theta_j\Big) + \hat{\Theta}_k\Big)^2 \\
\frac{\partial E_r}{\partial \hat{\omega}_{\hat{j}\hat{k}}} &= -\Big(t_{\hat{k}}^r - \Big(\sum_{j=1}^{o}\hat{\omega}_{j\hat{k}}y_j^r - \hat{\Theta}_{\hat{k}}\Big)\Big)y_{\hat{j}}^r = -\Big(t_{\hat{k}}^r - z_{\hat{k}}^r\Big)y_{\hat{j}}^r \\
\frac{\partial E_r}{\partial \omega_{\hat{i}\hat{j}}} &= -\sum_{k=1}^{m}\Big(t_k^r - z_k^r\Big)\hat{\omega}_{\hat{j}k}f'\Big(\sum_{i=1}^{n}\omega_{i\hat{j}}x_i^r - \Theta_{\hat{j}}\Big)x_{\hat{i}} \\
\Delta\hat{\omega}_{jk} &= \varepsilon\sum_{i=1}^{N}\frac{\partial E_r}{\partial \hat{\omega_{jk}}} \\
\Delta\omega_{ij} &= \varepsilon\sum_{i=1}^{N}\frac{\partial E_r}{\partial \omega_{ij}}
\end{aligned}$$

Una descripción algorítmica del algoritmo *Backpropagation* se presenta en el Algoritmo 4. La importancia de este procedimiento consiste en que, a medida que se entrena la red, las neuronas de las capas ocultas se organizan a sí mismas de tal modo que las distintas neuronas aprenden a reconocer distintas características del espacio total de entrada. Después del entrenamiento, cuando se les presente un patrón arbitrario de entrada que contenga ruido o que esté incompleto, las neuronas de la capa oculta de la red responderán con una salida activa si la nueva entrada contiene un patrón que se

asemeje a aquella característica que las neuronas individuales hayan aprendido a reconocer durante su entrenamiento. Y a la inversa, las unidades de las capas ocultas tienen una tendencia a inhibir su salida si el patrón de entrada no contiene la característica a reconocer, para la cual han sido entrenadas.

Algoritmo 4 Pseudocódigo del algoritmo *Backpropagation*

Entrada: Muestra de aprendizaje $\{(x^r, t^r)\}$ $\quad r = 1, \ldots, N$

1: Inicialización: Generar aleatoriamente los pesos y umbrales de la red.
2: Elegir una constante ε adecuada
3: **Para** cada uno de los N elementos de la muestra **hacer**
4: Calcular la salida de la red y_j
5: Calcular $\Delta\hat{\omega}_{jk}$ y ACTUALIZAR los pesos de la capa de salida según:
$\hat{\omega}_{jk} := \omega_{jk} + \Delta\hat{\omega}_{jk}$
6: Calcular $\Delta\omega_{ij}$
7: Actualizar los pesos de la capa oculta según:
$\omega_{ij} := \omega_{ij} + \Delta\omega_{ij}$
$k = 1, \ldots, m \qquad j = 1, \ldots, o \qquad i = 1, \ldots, N$
8: **fin Para**

4.6.4. Aplicación en el tratamiento de datos de miRNA

Dentro del estudio revisado en la sección anterior, [31, 60], junto al uso del clasificador k-NN, se entrena también un perceptrón multicapa (MLP) con las siguientes características:

- **Número de nodos de entrada: 25**. Como ya se indicó en la sección anterior, en el proceso de selección de características el número de miRNAs mejor clasificados es 25.
- **Número de nodos ocultos**: Van ajustándose entre 5 y 25. Finalmente, se entrena un MLP con 8 nodos en la capa oculta..
- **Número de nodos de salida: 2**. Corresponde al número de clases que se consideran en este problema, "normal" y "cáncer".
- **Tasa de aprendizaje:** $\varepsilon = 0.05$.
- **Algoritmo de aprendizaje**: *Backpropagation*.
- **Función de transferencia**: sigmoidea.

Con esta configuración se consideran las posibles combinaciones de métodos de selección de características vistos anteriormente con este clasificador (Figura 4.34), y de igual manera se compara con todos los clasificadores k-NN objeto de estudio en términos de *Accuracy*.

	IE	CP	CC	CS	IG	MI	SN
KNNE	92.7	92.4	91.7	90.0	91.7	93.0	90.8
KNNP	93.3	86.4	94.1	87.5	92.9	91.7	93.2
KNNP	92.2	85.9	93.2	87.7	90.6	90.6	90.9
KNNC	93.0	85.3	95.0	87.9	89.5	90.9	91.2
MLP	92.0	92.5	94.0	91.2	89.3	91.1	89.5

Figura 4.34: Comparativa entre clasificador MLP y variantes k-NN. Las columnas muestran los métodos de selección de características empleados.

Dentro de las posibles combinaciones de MLP con métodos de selección de características, una combinación de **MLP** con un método basado en la información mutua (MI) parece proporcionar la *Accuracy* más alta aunque, en conjunto, k-NNS + CC sigue siendo la combinación mas precisa.

Resultados del clasificador

En [60], se realiza un estudio comparativo de MLP junto con clasificadores SVM y NB trabajando sobre los datos de [35]. Sobre estos datos, se considera un subconjunto D_1 estructurado según la Figura 4.35. La comparativa de los resultados que cada uno de estos clasificadores proporciona queda recogido en la Figura 4.36 en términos del número de predicciones correctas que se realizan dentro de cada categoría.

Tejido	Número de muestras en estado ``normal''	Número de muestras en estado ``tumor''
Colon	5	10
Riñón	3	5
Próstata	8	6
Útero	9	10
Mama	3	6
Pulmón	4	6

Figura 4.35: Conjunto de muestras y estado que presentan.

Sobre este conjunto se impone la condición de que cada categoría (colon, riñón, próstata, útero, mama o pulmón) contenga, al menos, 3 muestras en estado "cáncer" y 5 en estado "normal". Como el número de muestras en estado "cáncer" es muy pequeño, se utiliza el método de validación *leave-one-out* para evaluar los resultados de los clasificadores sobre D_1. Puede observarse que los 3 clasificadores son exactos a la hora de clasificar los tejidos mamarios en su estado correspondiente.

Categoría (Tejido)	Número de muestras	MLP	SVM	NB
Colon	15	13	12	12
Riñón	8	8	8	7
Próstata	14	13	13	14
Útero	19	17	18	14
Mama	9	9	9	9
Pulmón	10	9	8	8

Figura 4.36: Número de predicciones correctas dentro de cada categoría (tejido), para cada clasificador considerado.

Bibliografía

[1] D.W. Aha, D. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991.

[2] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

[3] V. Ambros. The functions of animal microRNAs. *Nature*, 431(7006):350–355, 2004.

[4] E. Bandrés and J. García-Foncillas. Micro-ARN y cáncer. *Implicaciones clínicas de la investigación básica*, 8(5):248–253, 2009.

[5] D.P. Bartel. MicroRNAs: genomics, biogenesis, mechanism, and function. *Cell*, 116(2):281–297, 2004.

[6] D.P. Bartel. MicroRNAs: target recognition and regulatory functions. *Cell*, 136(2):215–233, 2004.

[7] A. Ben-Hur, C.S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch. Support vector machines and kernels for computational biology. *PLoS Comput Biol*, 4(10):e1000173, 2008.

[8] D. Betel, M. Wilson, A. Gabow, D.S. Marks, and C. Sander. The microRNA.org resource: targets and expression. *Nucleic acids research*, 36(suppl 1):D149–D153, 2008.

[9] J. Brennecke, A. Stark, R.B. Russell, and S.M. Cohen. Principles of microRNA–target recognition. *PLoS Biol*, 3(3):e85, 2005.

[10] M.J. Bueno, I. de Castro Pérez, M. de Cedrón Gómez, J. Santos, G.A. Calin, J.C. Cigudosa, C.M. Croce, J. Fernández-Piqueras, and M. Malumbres. Genetic and epigenetic silencing of microRNA-203 enhances abl1 and bcr-abl1 oncogene expression. *Cancer cell*, 13(6):496–506, 2008.

[11] Y. Cai, X. Yu, S. Hu, and J. Yu. A brief review on the mechanisms of miRNA regulation. *Genomics, proteomics & bioinformatics*, 7(4):147–154, 2009.

[12] E.J. Carmona Suárez. *Tutorial sobre Máquinas de Vectores Soporte (SVM).*

[13] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine learning*, 46(1-3):131–159, 2002.

[14] S.B. Cho and H.H. Won. Machine learning in dna microarray analysis for cancer classification. In *Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003-Volume 19*, pages 189–198. Australian Computer Society, Inc., 2003.

[15] C.Y. Chu and T.M. Rana. Small RNAs: regulators and guardians of the genome. *Journal of cellular physiology*, 213(2):412–419, 2007.

[16] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[17] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[18] N. Cristianini and J. Shawe-Taylor. An introduction to support vector machines, 2000.

[19] A.J. Enright, B. John, U. Gaul, T. Tuschl, C. Sander, and D.S. Marks. MicroRNA targets in drosophila. *Genome biology*, 5(1):R1–R1, 2004.

[20] R. Florez López and J.M. Fernández Fernández. *Las redes neuronales artificiales.* Netbiblo, La Coruña, 2008.

[21] J.A. Freeman Castro and D.M. Skapura González. *Redes neuronales: algoritmos, aplicaciones y técnicas de programación.* Addison-Wesley, Argentina, 1993.

[22] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning. Data Mining, Inference and Prediction.* Springer series in statistics Springer, Berlin, 2009.

[23] S. Griffiths-Jones, R.J. Grocock, S. Van Dongen, A. Bateman, and A.J. Enright. mirbase: microRNA sequences, targets and gene nomenclature. *Nucleic acids research*, 34(suppl 1):D140–D144, 2006.

[24] A. Grimson, K.K.H. Farh, W.K. Johnston, P. Garrett-Engele, L.P. Lim, and D.P. Bartel. MicroRNA targeting specificity in mammals: determinants beyond seed pairing. *Molecular cell*, 27(1):91–105, 2007.

[25] D.J. Hand and K. Yu. Idiot's bayes?not so stupid after all? *International statistical review*, 69(3):385–398, 2001.

[26] P.E. Hart. The condensed nearest neighbor rule. *Transactions on Information Theory*, 18:515–516, 1968.

[27] T. Hofmann, B. Schölkopf, and A.J. Smola. Kernel methods in machine learning. *The annals of statistics*, pages 1171–1220, 2008.

[28] P.W.C. Hsu, H.D. Huang, S.D. Hsu, L.Z. Lin, A.P. Tsou, C.P. Tseng, P.F. Stadler, S. Washietl, and I.L. Hofacker. miRNAMap: genomic maps of microRNA genes and their target genes in mammalian genomes. *Nucleic acids research*, 34(suppl 1):D135–D139, 2006.

[29] T.M. Huang, V. Kecman, and I. Kopriva. *Kernel based algorithms for mining huge data sets*, volume 1. Springer, Berlin, 2006.

[30] B. John, A.J. Enright, A. Aravin, T. Tuschl, C. Sander, and D.S. Marks. Human microRNA targets. *PLoS Biol*, 2(11):e363, 2004.

[31] K.J. Kim and S.B. Cho. Exploring features and classifiers to classify microRNA expression profiles of human cancer. In *Neural Information Processing. Models and Applications*, pages 234–241. Springer, Berlín, 2010.

[32] A. Krek, D. Grün, M.N. Poy, R. Wolf, L. Rosenberg, E.J. Epstein, P. MacMenamin, I. da Piedade, K.C. Gunsalus, and M. Stoffel. Combinatorial microRNA target predictions. *Nature genetics*, 37(5):495–500, 2005.

[33] D.T. Larose. K-nearest neighbor algorithm. *Discovering Knowledge in Data: An Introduction to Data Mining*, pages 90–106, 2005.

[34] R.C. Lee, R.L. Feinbaum, and V. Ambros. The c. elegans heterochronic gene lin-4 encodes small RNAs with antisense complementarity to lin-14. *Cell*, 75(5):843–854, 1993.

[35] J. Lu, G. Getz, E.A. Miska, E. Alvarez-Saavedra, J. Lamb, D. Peck, A. Sweet-Cordero, B.L. Ebert, R.H. Mak, and A.A. Ferrando. MicroRNA expression profiles classify human cancers. *nature*, 435(7043):834–838, 2005.

[36] D.G. Luenberger and Y. Ye. *Linear and nonlinear programming*, volume 2. Springer International Publishing, 1984.

[37] A. Lugo-Trampe and K.C. Trujillo-Murillo. MicroRNAs: reguladores clave de la expresión génica. *Medicina Universitaria*, 11(44):187–192, 2009.

[38] C. Ma, Y.F. Liu, and L. He. MicroRNAs-powerful repression comes from small RNAs. *Science in China Series C: Life Sciences*, 52(4):323–330, 2009.

[39] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[40] T.M. Mitchell. *Machine learning*. McGraw-Hill, New York, 1997.

[41] R. Mitra and S. Bandyopadhyay. Multimitar: a novel multi objective optimization based miRNA-target prediction method. *PloS one*, 6(9):e24583, 2011.

[42] M.W. Nabors and P. González-Barreda. *Introduction to botany*. Pearson Benjamin Cummings, New York, 2004.

[43] A. Navarro Ponz. *Análisis comparativo de la expresión de miRNAs en el desarrollo embrionario del colon, el cáncer colorectal y el linfoma de Hodgkin*. PhD thesis, Facultad de medicina, Universidad de Barcelona, 2008.

[44] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[45] M.N. Poy, J Hausser, M Trajkovski, M Braun, S Collins, P. Rorsman, M. Zavolan, and M. Stoffel. mir-375 maintains normal pancreatic α-and β-cell mass. *Proceedings of the National Academy of Sciences*, 106(14):5813–5818, 2009.

[46] V.L. Robinson. Rethinking the central dogma: Noncoding RNAs are biologically relevant. *Urologic Oncology: Seminars and Original Investigations*, 27(3):304–306, 2009.

[47] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[48] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

[49] B. Sangro. Micro-ARN. posibles implicaciones terapéuticas en las enfermedades hepáticas. *Implicaciones clínicas de la investigación básica*, 10(8):170–174, 2011.

[50] P. Sethupathy, B. Corda, and A.G. Hatzigeorgiou. Tarbase: A comprehensive database of experimentally supported animal microRNA targets. *Rna*, 12(2):192–197, 2006.

[51] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, United Kingdom, 2004.

[52] B.W. Silverman and M.C. Jones. E. Fix & J.L. Hodges (1951): An important contribution to nonparametric discriminant analysis and density estimation: Commentary on Fix and Hodges (1951). *International Statistical Review / Revue Internationale de Statistique*, 57(3):233–238, 1989.

[53] M. Steinbach and P.N. Tan. knn: k-nearest neighbors. In W. Xindong and K. Vipin, editors, *The top ten algorithms in data mining*, chapter 8, pages 151–161. CRC Press, New York, 2009.

[54] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, New York, 2013.

[55] V.N. Vapnik and V. Vapnik. *Statistical learning theory*, volume 1. Wiley, New York, 1998.

[56] G. Vázquez-Ortiz, P. Pina-Sanchez, and M. Salcedo. Great potential of small RNAs: RNA interference and microRNA. *Revista de investigación clínica; órgano del Hospital de Enfermedades de la Nutrición*, 58(4):335–349, 2005.

[57] M.C. Vella, K. Reinert, and F.J. Slack. Architecture of a validated microRNA: target interaction. *Chemistry & biology*, 11(12):1619–1623, 2004.

[58] D.L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, pages 408–421, 1972.

[59] H.C. Wu. The Karush-Kuhn-Tucker optimality conditions in multiobjective programming problems with interval-valued objective functions. *European Journal of Operational Research*, 196(1):49–60, 2009.

[60] R. Xu, J. Xu, and D.C. Wunsch. MicroRNA expression profile based cancer classification using default ARTMAP. *Neural Networks*, 22(5):774–780, 2009.

[61] M. Yousef, S. Jung, A.V. Kossenkov, L.C. Showe, and M.K. Showe. Naïve bayes for microRNA target predictions-machine learning for microRNA targets. *Bioinformatics*, 23(22):2987–2992, 2007.

[62] M. Yousef, M. Nebozhyn, H. Shatkay, S. Kanterakis, L.C. Showe, and M.K. Showe. Combining multi-species genomic data for microRNA identification using a naive bayes classifier. *Bioinformatics*, 22(11):1325–1334, 2006.