Implementació d'una eina de predicció de dianes de miRNA basat en algorismes de Machine Learning

Jordi Carrere Molina
Màster de Bioinformàtica i Bioestadística
Machine Learning

Consultor: Albert Pla Planas
Professor: Àlex Sànchez Pla

24/05/2017

FITXA DEL TREBALL FINAL

Títol del treball:	*Implementació d'una eina de predicció de dianes de miRNA basat en algorismes de Machine Learning*
Nom de l'autor:	*Jordi Carrere Molina*
Nom del consultor/a:	*Albert Pla Planes*
Nom del PRA:	*Àlex Sànchez Pla*
Data de lliurament (mm/aaaa):	*24/05/2017*
Titulació o programa:	*Màster de Bioinformàtica i Bioestadística*
Àrea del Treball Final:	*Machine Learning*
Idioma del treball:	*Català*
Paraules clau	*MiRNA, target, Machine learning*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

Els miRNA són cadenes curtes de RNA no codificant, d'aproximadament 22 nucleòtids, amb funció reguladora de l'expressió gènica a nivell post-transcripcional. Es coneixen uns 1500 miRNAs en humans que es calcula que regulen el 30% dels gens. La identificació de les dianes de cada miRNA és essencial per entendre la seva funció biològica. La predicció de les diferents dianes d'un miRNA *in silico* és fonamental per estalviar temps i recursos per, posteriorment, validar-les experimentalment. Diferents softwares fan aquestes prediccions gràcies a models generats a partir de regles basades en l'observació empírica. En els darrers temps, s'han aplicat algorismes de *Machine Learning* per modelar la interacció entre el miRNA i la seva diana, amb grans resultats d'exactitud. El *Machine Learning* és un conjunt d'algorismes que permeten detectar patrons en gran quantitat de dades, generant models que permetin classificar o predir nous exemples. En aquest treball es presenta l'eina miRNAforest, que mitjançant l'algorisme *Random Forest,* modela la unió del miRNA i les seves dianes per a classificar noves possibles dianes d'un miRNA donat. miRNAforest és capaç de fer prediccions de dianes de miRNA amb una exactitud del 85.78%, sensibilitat del 86.93% i especificitat del 84.66%.

Abstract (in English, 250 words or less):

miRNA are short non-coding RNA, approximately 22 nucleotides long, with regulatory function of gene expression at post-transcriptional level. About 1500 human miRNA are known, that it is estimated to regulate 30 % of human genes. The identification of miRNA targets is essential to understanding its biological function. Prediction of miRNA targets *in silico* is a key method to save time and resources to subsequently validate them experimentally. Different softwares can do these predictions applying rule based algorithms. In a recent time, some machine learning algorithms have been applied to model the interaction between miRNA and its target, obtaining great accuracy results. *Machine Learning* is a method able to recognise patterns in large amounts of data and device a model to classify or predict new data. This thesis presents the tool miRNAforest that models the union between miRNAs and their targets by Random Forest algorithm to classify a new possible target of a given miRNA. miRNAforest is able to predict miRNA targets with an accuracy of 85.78%, 86.93% of sensitivity and 84.66% of specificity.

Índex

Llista de figures

Llista de taules

1. Introducció

1.1 Context i justificació del Treball

Des del descobriment de l'estructura del DNA per James Watson i Francis Crick el 1953, s'ha intentat entendre el funcionament d'aquesta molècula, definida com la base de la vida. El 1990, un consorci internacional va iniciar el “Projecte Genoma Humà”, que va concloure el 2003. Aquest macro projecte tenia la finalitat d'obtenir la seqüència completa del genoma humà, per posteriorment trobar els gens que el componen, així com altres seqüències i les seves respectives funcions. Però el coneixement del nostre genoma no ens permet explicar el per que una neurona i un hepatòcit d'un mateix individu, amb un genoma idèntic, tenen una morfologia i funció tan diferents. La clau la trobem en com les cèl·lules de diferents teixits utilitzen de forma diferent la informació continguda en el seu nucli. L'expressió gènica de cada cèl·lula està, i ha d'estar, extremadament regulada, tant de forma intrínseca com extrínseca. És en aquest procés on l’epigenètica té un paper fonamental. L'epigenètica es defineix com els canvis heretables en l'expressió gènica que ocorren sense una alteració en la seqüència de nucleòtids del DNA [1]. Aquesta regulació de l'expressió gènica s’ha descrit tradicionalment per dos mecanismes (Figura 1), la metilació del DNA [2] i les modificacions en les histones (metilació, acetilació, ubuquitinació, fosforilació i sumoilació) [3], però altres marques epigenètiques són actualment objecte d’estudi, com ara la disposició dels nucleosomes [4] i l'activitat dels RNA no codificants (ncRNA:microRNA i RNA antisentit, entre d'altres) [5].

Els microRNAs (miRNA) són cadenes curtes de ncRNA, d’aproximadament 22 nucleòtids, que regulen l’expressió gènica a nivell post-transcripcional [6]. Els gens que codifiquen per als miRNA són molt més llargs que el miRNA madur. El transcript es plega sobre si mateix formant una estructura en forquilla (pri-miRNA), estabilitzada per unions Watson-Crick entre els dos braços de la forquilla. Aquest pri-miRNA és processat fins a obtenir un miRNA madur capaç de regular l’expressió gènica a nivell post-transcripcional (Figura 2).

Cada miRNA és capaç de reconèixer un o més RNA missatgers, regulant-ne la seva traducció. La identificació de les dianes de cada miRNA és fonamental per entendre la seva funció biològica. Actualment es coneixen al voltant de 1500 miRNAs en humans, que es creu que regulen el 30% dels gens [7]. Els miRNA actuen en l’expressió gènica a nivell de transcript, ja sigui reprimint la seva traducció (síntesi de la proteïna codificada en el transcript), com accelerant-ne

la seva degradació (figura 4). Per du a terme la seva funció, els miRNAs s'uneixen amb una proteïna de la família Argonaute (AGO) per formar el núcli del complex de silenciament (miRISCs) [8]. Aquest complex s'uneix a un mRNA diana, depenent de la seqüència del miRNA i d'una part del mRNA, en general l'extrem 3'UTR.

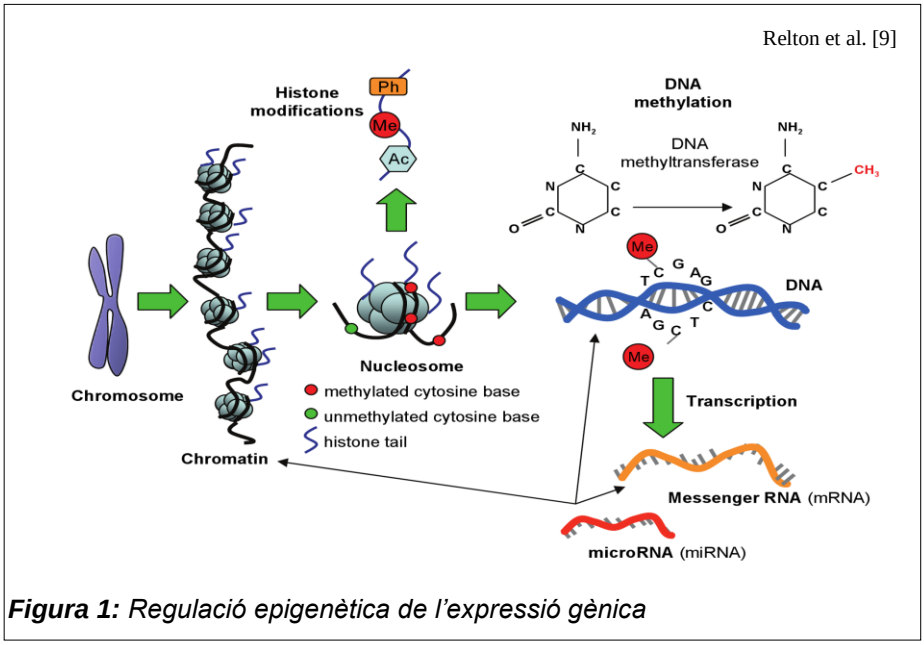


Figura 1: *Regulació epigenètica de l'expressió gènica*

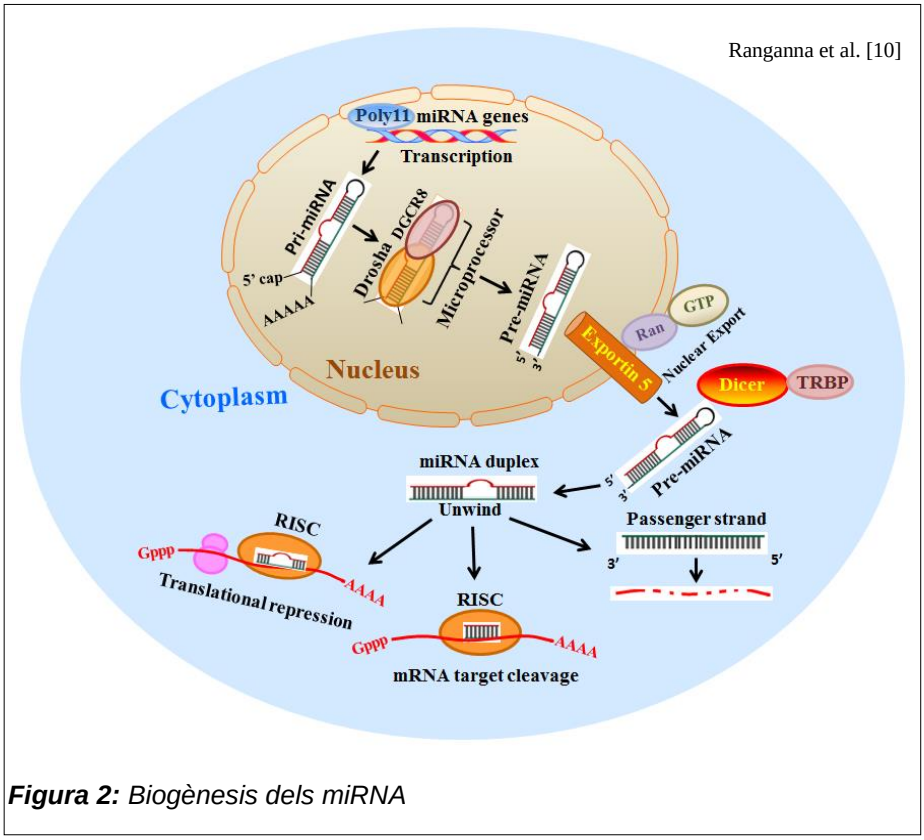


Figura 2: *Biogènesis dels miRNA*

1.1.1. Descriptors per a la predicció de dianes

El descobriment de les dianes de cada miRNA es fa gairebé impossible a nivell experimental tant per la complexitat com pels costos econòmics. És per això que s'han desenvolupat diferents eines de predicció computacional d'aquestes dianes. Aquestes eines utilitzen diversos algorismes per identificar si un mRNA és una potencial diana d'un miRNA determinat a partir de característiques de la interacció entre les dues molècules. Aquestes característiques (descriptors) s'han detectat mitjançant l'observació i l'anàlisi estadístic. Els descriptors més comuns en els diferents algorismes de predicció de diana es descriuen a continuació:

Seqüència llavor: Es considera seqüència llavor als primers 8 nucleòtids de l'extrem 5' del miRNA. Existeixen 2 tipus de seqüències llavor, les canòniques i les no canòniques. Les primeres tenen una alta complementarietat amb el mRNA (figura 3), mentre que les no canòniques no tenen aquest grau de complementarietat i necessiten aparellaments compensatoris a altres regions del mRNA. Alguns tipus d'aparellaments canònics segons el grau de coincidència són:

- *6mer*: aparellament perfecte WC (Watson-Crick) dels nucleòtids 2-7 del miRNA amb la regió diana del mRNA.
- *7mer-m8*: aparellament perfecte WC dels nucleòtids 2-8 de la llavor del miRNA.
- *7mer-A1*: aparellament perfecte WC dels nucleòtids 2-7 de la llavor del miRNA i una A en la posició 1.
- *8mer*: aparellament perfecte WC dels nucleòtids 2-8 de la llavor i una A en la posició 1.

Peterson et al. [11]

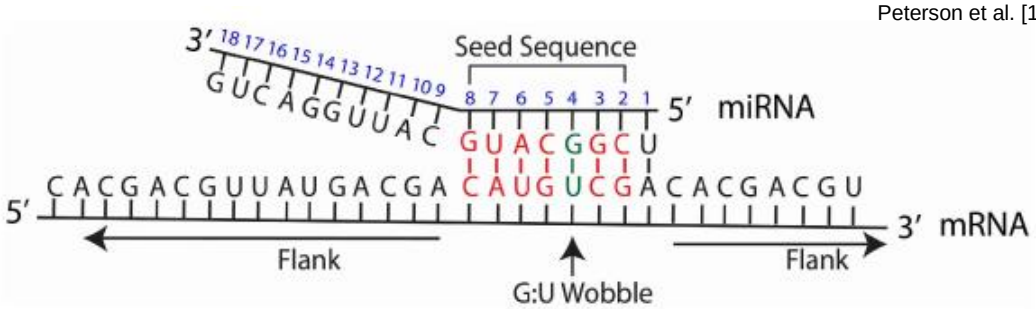


Figura 3: *Unió canònica de la seqüència llavor del miRNA amb la seva diana. En alguns casos, alguns aparellaments G:U (G:U wobble) són tolerats*

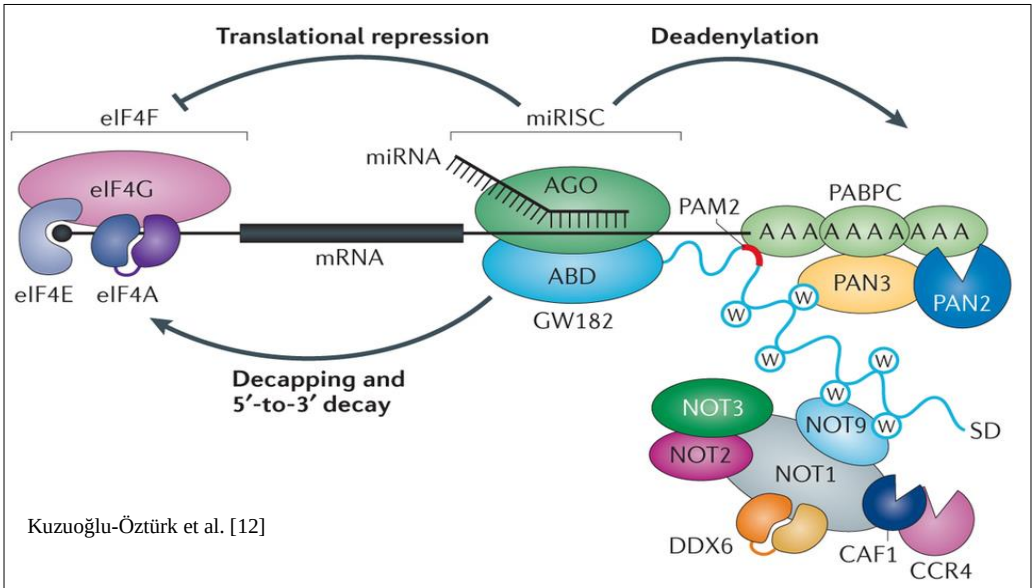


Figura 4: *Mecanismes de repressió del mRNA per part d'un miRNA*

Conservació: Fa referència al manteniment d'una seqüència al llarg de l'evolució. S'observa una major conservació en la seqüència llavor del miRNA que en la resta de la seva seqüència [13]. En alguns casos, l'aparellament de seqüències conservades a l'extrem 3' del miRNA pot compensar aparellaments incorrectes en la seqüència llavor. Alguns softwares de predicció de dianes fan un anàlisi de conservació de la regió 3'UTR, la 5'UTR, el miRNA o combinacions d'aquestes 3.

Energia lliure: L'energia lliure de Gibbs [14] permet mesurar el grau d'estabilitat de la unió miRNA:mRNA. Si aquesta unió es preveu estable, augmenten les probabilitats que el mRNA objecte d'estudi sigui una diana real del miRNA.

Accessibilitat: És una mesura de la facilitat amb que el miRNA pot accedir i hibridar amb la potencial diana [11]. Els mRNA adopten una estructura secundària un cop transcrits. Aquesta estructura pot dificultar l'accés a la seqüència diana per part de la seqüència llavor. Un cop la llavor s'uneix a la diana, el mRNA va perdent la seva estructura secundària a mesura que es van aparellant. L'energia necessària per a culminar aquest procés permet avaluar la versemblança de la diana.

Descriptors in-situ: A més de la seqüència llavor, altres descriptors important es poden trobar en altres regions del transcripts, sobretot en l'extrem 3'UTR. La seqüència del miRNA es divideix en 3 regions: la regió 5' (anomenada regió llavor), la regió 3' i la regió total. En aquestes tres regions, diferents característiques poden ser avaluades, com l'energia lliure o el

nombre d'aparellaments G:C, A:U o G:U. Fora la regió llavor es poden trobar mecanismes compensatoris que permetin la unió del miRNA a mRNA amb una seqüència llavor de dianes no canòniques. Alguns softwares de predicció tenen en compte la ubicació espacial de la seqüència llavor dins l'extrem 3'UTR.

1.1.2. *Software* de predicció

Existeixen diferents eines de predicció, tant softwares lliures executables de forma local, com aplicacions web. Cadascuna utilitza una sèrie de descriptors diferents i un algorisme propi. En la taula 1 es fa una comparativa dels algorismes més importants per a la predicció de dianes de miRNA, que es resumeixen a continuació:

TargetScan [15]
Permet la predicció de dianes tant conservades com no conservades. Pel que fa a les dianes conservades, en primer lloc busca aparellaments 8mer, 7mer-m8 o 7mer-A1. Totes les potencials dianes del 3'UTR serveixen per a calcular la probabilitat que sigui una diana conservada (P_{CT}), i el "context++ scores", que calcula la probabilitat que una diana determinada sigui una diana real. Aquest valor es calcula de diferents descriptors, com aparellaments compensatoris a 3', contingut local en AU i la ubicació de la diana dins la seqüència de l'extrem 3'UTR del transcript.

miRANDA-mirSVR [16]
miRANDA identifica possibles dianes en tres passos: (1) Es busquen aparellaments WC entre les seqüències de miRNA i mRNA introduïdes per l'usuari. (2) De cada possible aparellament trobat en el punt anterior, es calcula l'energia lliure. Aquells que tinguin una energia lliure inferior a un llindar, passa al següent pas. (3) L'última característica per tal de catalogar un mRNA com a diana o no és la conservació. Un cop identificats els llocs diana, mirSVR les puntua amb una regressió de vector de suport (SVR) de forma similar als SVM, donant una mesura de l'efecte d'un miRNA en l'expressió del mRNA.

PITA [17]
Utilitza com a descriptor principal l'accessibilitat al lloc diana, basant-se en l'observació que els llocs dianes estan preferentment ubicats en regions accessibles de l'UTR. Identifica dianes potencials per aparellament de llavor, i després considera l'accessibilitat calculant l'energia lliure basada en la diferència entre el guany d'energia per la formació del dúplex miRNA:mRNA i el cost de separar la diana per fer-la accessible. Finalment es combinen els

valors d'accessibilitat del diferents possibles aparellaments del miRNA amb un UTR determinat, obtenint una puntuació d'interacció total.

DIANA-microT-CDS [18]
Utilitza *Machine learning* per identificar descriptors a partir de dades de PAR-CLIP (photoactivatable-ribonucleoside-enhanced crosslinking and immunoprecipitation). D'aquesta manera, l'algorisme aprÈn descriptors associats amb miRNA que tenen el lloc d'unió tant en el CDS com en el 3'UTR. Per altre banda, dades d'arrays d'expressió permeten aprendre la contribució de diferents llocs d'unió en un transcript diana. Per les dues regions (CDS i 3'UTR), els descriptors més rellevants són la eficiència de la unió, la distància a l'extrem més proper de la regió, la distància a un lloc d'unió adjacent, l'energia lliure, la conservació i el contingut en AU.

RNAhybrid [19]
Considera l'energia lliure entre el miRNA i un mRNA amb una regió llavor definida per l'usuari. Ofereix una gran nombre de opcions avançades a l'usuari. Cal introduir la seqüencia del miRNA i del 3'UTR del mRNA. Està destinada a usuaris avançats ja que requereix l'entrada de les dues seqüències en format FASTA i permet la manipulació de múltiples paràmetres específics d'aquesta eina.

mirTarget2 [20]
Prediu dianes amb SVM i descriptors extrets d'un gran microarray com a dataset d'aprenentatge. Aquest mètode confirma la utilitat de molts dels predictor usats en la predicció de dianes i n'identifica de nous. Utilitza com a descriptors la conservació de la seqüència llavor, el correcte aparellament en les posicions 2-8 de la seqüència llavor, contingut de bases en les regions adjacents a lloc d'unió de la llavor, estructura secundària i ubicació dins el 3'UTR.

rna22-GUI [21]
Identifica illes diana mitjançant descobriment de patrons ("pattern discovery") i avalua l'energia lliure de les illes diana i els miRNA candidats. Permet buscar per miRNA, gene ID, transcript ID, o nom del gen.

TargetMiner [22]
Es tracta d'un altre software basat en un classificador SVM per identificar potencials llocs llavor entre un miRNA introduït per l'usuari i un mRNA seleccionat. Es poden carregar múltiples miRNAs i mRNAs en un fitxer per a ser avaluats.

ComiR [23]
És un algoritme d'*Ensembl prediction*. Prediu si un mRNA està regulat per un o més (o cap) miRNA. Utilitza l'expressió de miRNA combinat amb altres algorismes de predicció (miRanda, PITA, TargetScan i mirSVR), per d'un algorisme d'*ensembl prediction*. La puntuació dels quatre algorismes es combinen amb un SVM per definir la predicció.

deepTarget [24]
Es tracta d'una eina basada en *Machine Learning*. Utilitza dos autoencoders per a modelar les seqüències del miRNA i del mRNA respectivament. La segona capa conté una xarxa neuronal recurrent (RNN) que modela la interacció entre les dues seqüències.

	Seed match	Conservació	Energia lliure	Accessibilitat	Altres
TargetScan	✓	✓			· aparellament compensatori a 3' · contingut AU · ubicació de la diana
miRANDA-mirSVR	✓	✓	✓	✓	· contingut AU · ubicació de la diana · llargada de l'UTR
PITA	✓	✓	✓	✓	· abundància de dianes
DIANA-microT-CDS	✓	✓	✓	✓	· *Machine learning* · abundància de dianes · contingut AU
RNAhybrid	✓		✓		· abundància de dianes
mirTarget2	✓	✓	✓	✓	· composició regions confrontants a la llavor · estructura secundària · ubicació de la diana
RNA22-GUI	✓		✓		
TargetMiner	✓	✓	✓	✓	· abundància de dianes
ComiR					· SVM · Ensembl prediction miRanda PITA TargetScan mirSVR
deepTarget					· Autoencoder · RNN

Taula 1: *Sumari dels principals algorismes de predicció de dianes*

1.1.3. Algorismes de *Machine Learning*

En aquest context, les tècniques de *machine learning* poden aportar noves solucions a aquest problema. El *machine learning* és una branca de la intel·ligència artificial que crea sistemes amb la capacitat d'identificar patrons complexes en grans quantitats de dades de forma automàtica. A partir de

l'anàlisi de les dades, es generen models automàticament, mitjançant algorismes que aprenen de forma iterativa, identificant patrons i característiques difícilment detectables mitjançant l'observació i l'anàlisi estadístic. L'elecció de l'algorisme i els predictors que el defineixen són claus per a obtenir prediccions fiable.

Els diferents algorismes de *Machine Learning* es poden classificar entre aprenentatge supervisats i aprenentatge no supervisats. L'aprenentatge supervisat consisteix en modelar una sèrie de predictors (l'equivalent a les variables independents en regressió lineal) respecte a una variable de sortida (l'equivalent a la variable dependent en regressió lineal). Durant el procés d'aprenentatge, s'utilitzen exemples amb un valor de sortida, numèric o categòric, conegut (dades d'aprenentatge) per generar un model que ens permeti predir el valor de sortida de nous exemples. En finalitzar la fase d'aprenentatge, s'avalua el model comparant les prediccions efectuades pel model dels exemples continguts en el set de dades d'avaluació amb els valors coneguts de sortida. En l'aprenentatge no supervisat no tenim cap coneixement a priori de la característica que volem analitzar. Són algorismes que permeten obtenir descriptors de les dades, detectar patrons,o relacions entre les dades del data set o agrupar els exemples de les nostres dades segons el seu grau de semblança, és a dir, permeten generar classificacions. Aquest últim procés rep el nom de *clustering.* Un cas especial són els autoencoders, que es poden classificar com a auto-supervisats, ja que tot i no requerir de informació de classe per a l'aprenentatge, gairebé sempre s'utilitzen per a tasques de classificació supervisada, utilitzant com a dades de sortida les mateixes que les d'entrada. Està fora de l'objectiu d'aquest treball aprofundir en l'aprenentatge no supervisat. Alguns dels algorismes d'aprenentatge automàtic supervisat més importants es descriuen a continuació:

Naive Bayes [25]

És un mètode simple per aplicar el teorema de Bayes (o probabilitat condiciona) als problemes de classificació que assumeix la independència entre els predictors. Tot i no ser l'únic mètode de *machine learning* que utilitza mètodes bayesians, és el més comú. A partir de dades conegudes, genera un model que calcula la probabilitat que un nou exemple pertanyi a un grup o un altre. L'algorisme calcula la probabilitat que un exemple determinat amb les característiques f_1,..., f_n pertanyi a cada possible grup, i el classifica en el grup més probable:

$$\text{classificador}(f_1, \ldots, f_n) = \underset{c}{\text{argmax}}\ p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c)$$

Xarxes Neuronals Artificials (ANN) [26]

Les ANN modelen la relació entre un grup de senyals d'entrada i una senyal de sortida (figura 5), utilitzant un model inspirat en el funcionament de les neurones i les seves interaccions en els cervells animals. Diferents neurones artificials (o nodes) s'interconnecten formant una xarxa per solucionar problemes d'aprenentatge. Les neurones de la capa amagada reben informació de les neurones de la capa visible, activant-se o no segons una funció d'activació. Algunes de les funcions d'activació utilitzades en les xarxes neuronals són la funció ReLU o la funció tanh. Existeixen diferents tipus de xarxes neuronals, com les Convolutional Neural Networks (CNN) o les Recurrent neural networks (RNN).

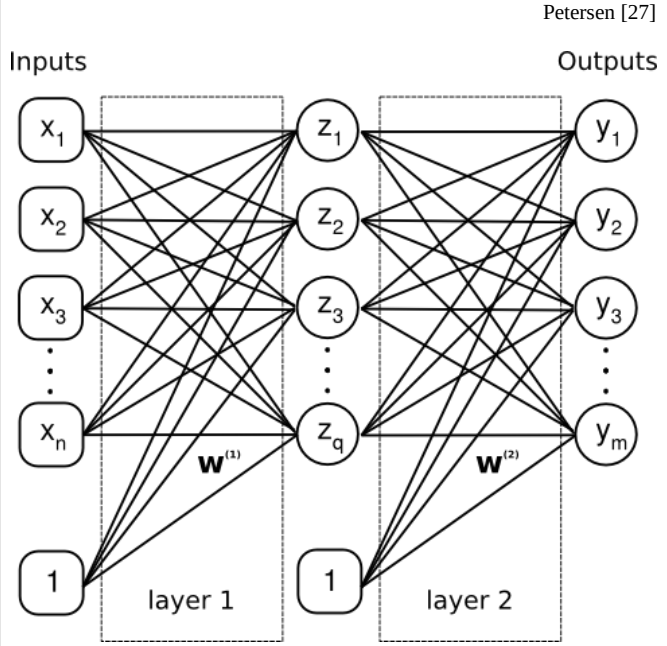


Figura 5: *Esquema bàsic d'una xarxa neuronal artificial (ANN)*

k-Nearest Neighbour (kNN) [28]

És un dels algorismes més simple de *machine learning*, però segueix en ús en alguns casos degut a la seva efectivitat i ràpida fase d'aprenentatge. Utilitza la informació dels k veïns més propers per a classificar un exemple no classificat, on k és el nombre casos que es prenen per classificar l'exemple problema. El valor de k pot ser determinant per a obtenir un bon model. Existeixen diferents estratègies per escollir el valor de k, des d'utilitzar $k = \sqrt{n}$, on n és el nombre d'exemples del set de dades d'entrenament, com utilitzar un valor gran i ponderant els vots segons la distància amb l'exemple problema (figura 6).

Logistic Regression [29]

És un cas especial de l'anàlisi de regressió en que la variable depenent és dicotòmica. Per tant, és una tècnica multivariable de dependència ja que pretén estimar la probabilitat que succeeixi un esdeveniment en funció de la dependència d'altres variables independents. És un cas de model lineal generalitzat (GLM) que modela la probabilitat d'un esdeveniment en funció d'altres factors, utilitzant la funció logit com a funció d'enllaç (figura 7).

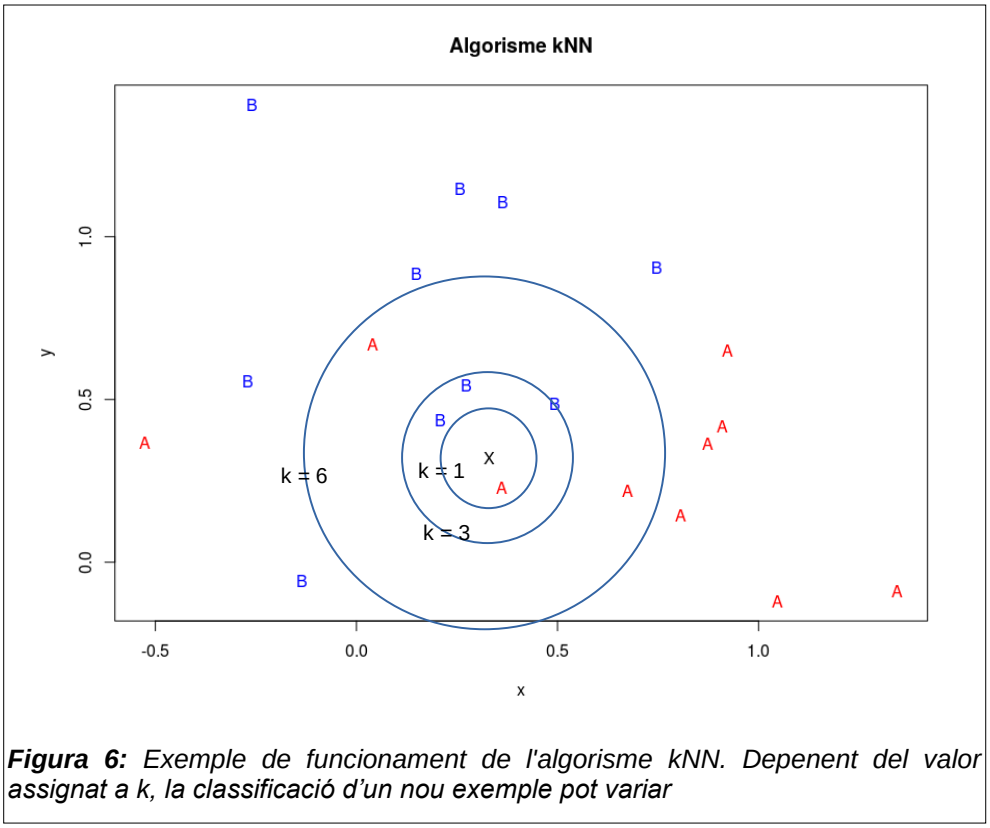


Figura 6: *Exemple de funcionament de l'algorisme kNN. Depenent del valor assignat a k, la classificació d'un nou exemple pot variar*

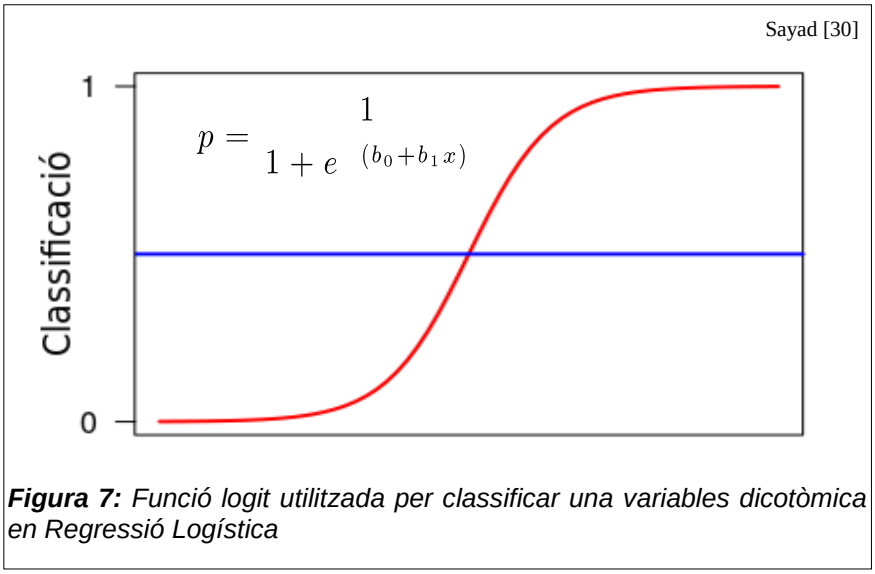


Figura 7: *Funció logit utilitzada per classificar una variables dicotòmica en Regressió Logística*

Decision Tree [31]

Es tracta d'un mètode per a la presa de decisions adequades tenint en compte varies alternatives possibles d'acció. Mitjançant una estructura en arbre, modelen les relacions entre les característiques i els potencials resultats, per tal de classificar nous exemples. Són models precisos, estables i fàcils d'interpretar ja que es basen en unes regles de decisió representades en arbres (figura 8).

Random forest [32]

Es tracta d'un mètode que genera diversos arbres de decisió a partir d'un nombre determinat de característiques, escollides de forma aleatòria, de manera que cada arbre no conté totes les variables de les nostres dades, però si que cada variable està present en diferents arbres del bosc. Un cop generat el conjunt d'arbres, el model utilitza el vot de cadascun per a predir nous exemples (figura 9).

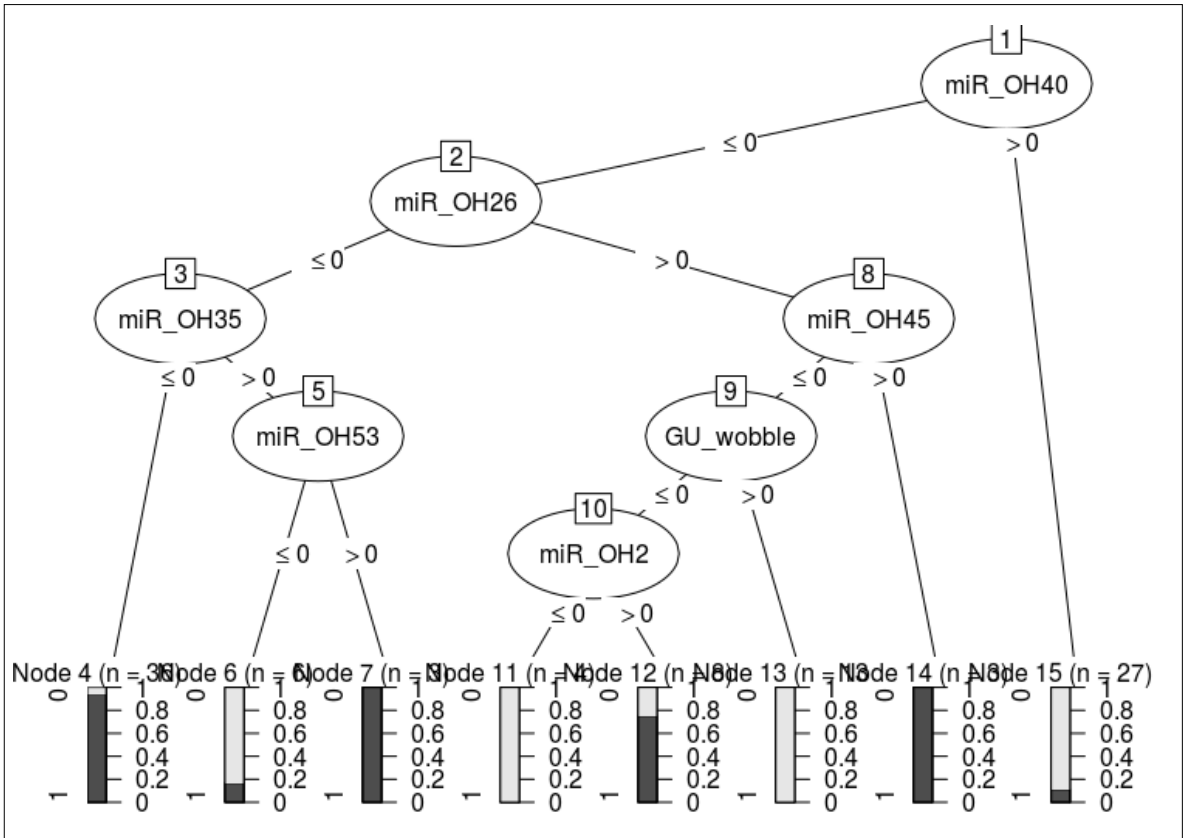
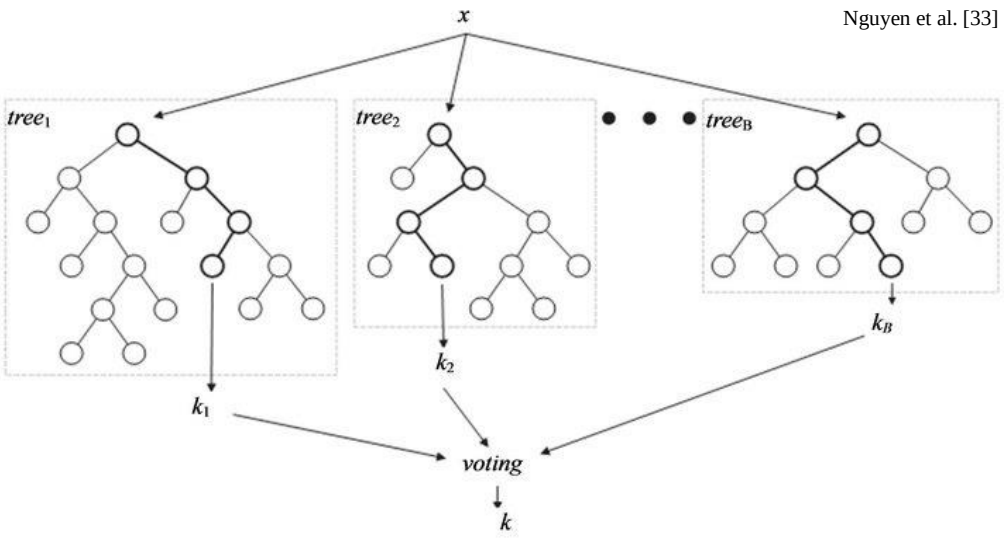


Figura 8: *Exemple d'arbre de decisió. L'algoritme genera una sèrie de condicions que s'han de cumplir en l'ordre indicat fins a arribar a un node final*

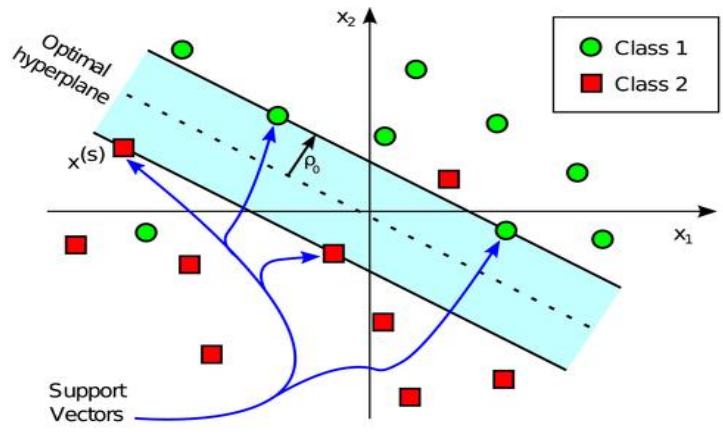


Nguyen et al. [33]

Figura 9: *Esquema bàsic de l'algorisme Random Forest*

Support vector machine (SVM) [34]

Les màquines de suport vectorial o màquines de vectors de suport, són un conjunt d'algorismes d'aprenentatge supervisat, enfocats a la resolució de problemes tant de classificació com regressió. Una SVM és un model que representa els punts de la mostra en l'espai, increment la dimensionalitat de l'espai de cerca, separant les diferents classes en l'espai mitjançant hiperplans de separació definits com els vectors entre els dos punts, de dues classes, més propers anomenats vectors de suport (figura 10). Més formalment, una SVM construeix un hiperplà o un conjunt d'hiperplans en l'espai de dimensionalitat molt alta per a crear particions el més homogènies possibles a cada cantó dels hiperplans.



Ruiz-Gonzalez R et al. [35]

Figura 10: *Exemple d'hiperplà definit per diferents vectors de suport d'una SVM*

Boosting [36]

És un algorisme d'*ensemble learning* que combina la predicció de diversos estimadors dèbils per generar un estimador robust. Cada estimador té un pes en funció de l'exactitud de les seves prediccions. Es basa en l'observació que és més fàcil trobar un conjunt de regles generals dèbils que una sola regla general robusta. A cada iteració, l'algorisme genera una regla de predicció dèbil a partir d'un subset de les dades d'aprenentatge. Després de moltes iteracions l'algorisme combina aquestes regles dèbils en una única regla de predicció, que serà molt més acurada que qualsevol de les regles dèbils (figura 11).

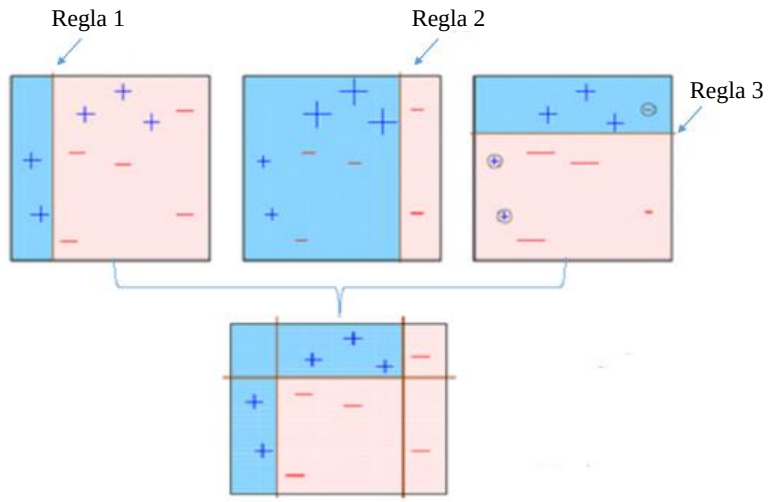


Figura 11: *Exemple de Boosting. Cada regla té, individualment, una capacitat dèbil de separar els diferents exemples. En utilitzar totes les regles en conjunt s'obté un model capaç de classificar els nous casos de forma acurada*

Dins l'aprenentatge automàtic, existeix un conjunt d'algorismes anomenats *Feature Learning,* un conjunt de tècniques que extreuen les característiques més rellevants de les dades a estudi. Permeten detectar les variables o combinació de variables que defineixen les característiques més rellevants de les dades objecte d'anàlisi, per tant, són mètodes de reducció dimensional. A continuació es descriuen algunes de les tècniques més importants de reducció dimensional:

Principal component analysis (PCA) [37] **i**
Independent component analysis (ICA) [38]

No són pròpiament tècniques de *Feature learning*, però si són dos tipus d'anàlisi importants en el processat de dades. Són tècniques estadística de reducció de la dimensionalitat mitjançant la combinació ponderada de predictors. En aquest sentit, actúen de manera similar al *Feature Learning*, ja que extreuen les característiques més rellevants de les dades. La principal

diferència entre aquestes dues tècniques és que el ICA permet l'ús de variables amb distribució no gaussiana, mentre que per tal de fer un PCA és necessari que les variables d'entrada tinguin distribució normal. Les noves variables capturen la variància de les dades, en el cas del PCA, el primer component principal concentra el percentatge més gran de variància de les nostres dades, el segon component principal conté el segon percentatge més gran de variància, i així successivament. D'aquesta manera es poden utilitzar els n primers components principals, o components independents, sabent amb quin percentatge de variància estem treballant. A més de la reducció en el nombre de descriptors, aquestes tècnica permet reduir el soroll de les dades originals.

Xarxes Neuronals Artificials (ANN)

Les ANN són algorismes que poden ser utilitzades per a *Feature Learning,* ja que permeten reduir la dimensionalitat amb una capa central amb pocs nodes [39]. A continuació es resumeixen algunes de les ANN utilitzades en *Feature Learning:*

- Restricted Boltzmann Machines (RBM): Les RBM són algoritmes de reducció dimensional, classificació, regressió i *feature learning* [40]. Són un tipus especialitzat de xarxa neuronal artificial de dues capes que constitueixen els blocs de les deep-belief networks, que s'explicaran a continuació.

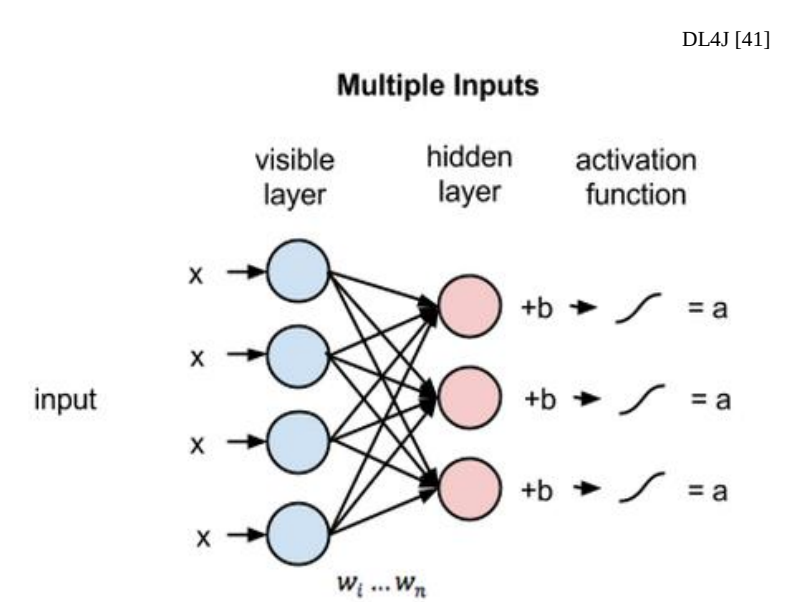


Figura 12: *Estructura d'una Restricted Boltzmann Machines*

En la figura 12, en la capa visible, els valors de les diferents variables es multipliquen per un pes (w_i) i es suma un biaix (b) en cada node de

la capa oculta. Al valor resultant se li aplica una funció d'activació, de manera que si es supera un llindar el node dona un valor de sortida d'activació.

- Deep Belief neural networks: Són xarxes multicapa, en que cada capa és una RBM [42] (figura 13). En el primer pas, la primera capa oculta aprèn descriptors de la capa visible, seguint els mateixos passos que una RBM. En el segon pas, la capa oculta anterior actuarà com a capa visible per entrenar una nova capa oculta, i així successivament fins a completar tota la xarxa. A cada nova capa oculta, la xarxa és capaç d'aprendre característiques més complexes.
- Autoencoders: Són xarxes neuronals artificials utilitzades en *feature learning* [43]. Tenen una estructura similar a un perceptró multicapa, amb la diferència que la capa de sortida conté el mateix nombre de nodes que la capa d'entrada (figura 14). La funció dels autoencoders és aprendre els descriptors més rellevants d'un set de dades, amb la finalitat de reduir la dimensionalitat. Si tenim en compte que un autoencoder té el mateix nombre de nodes d'entrada que de sortida, és evident que ha de ser una capa oculta la que ens permet reduir la dimensionalitat.

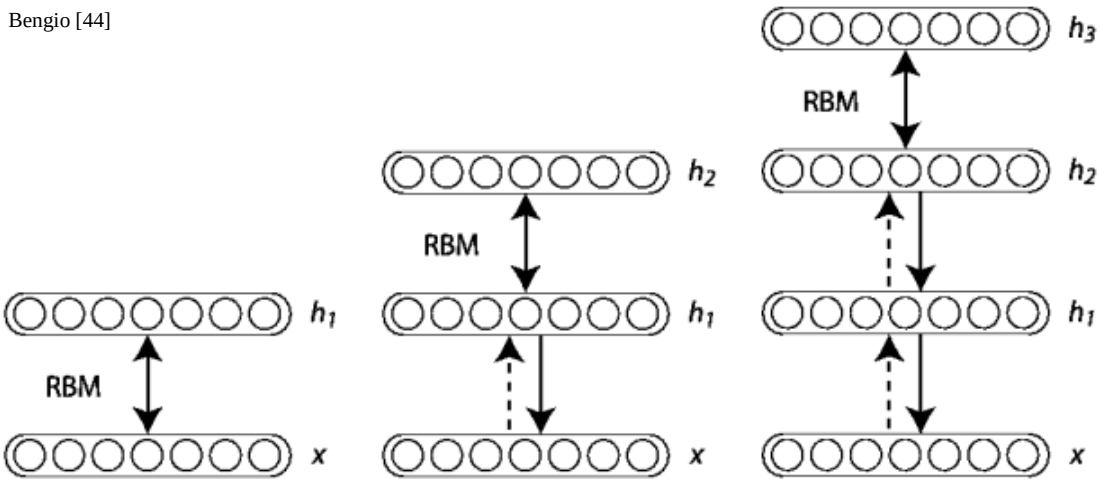


Figura 13: *Arquitectura d'una Deep Belief neural networks*

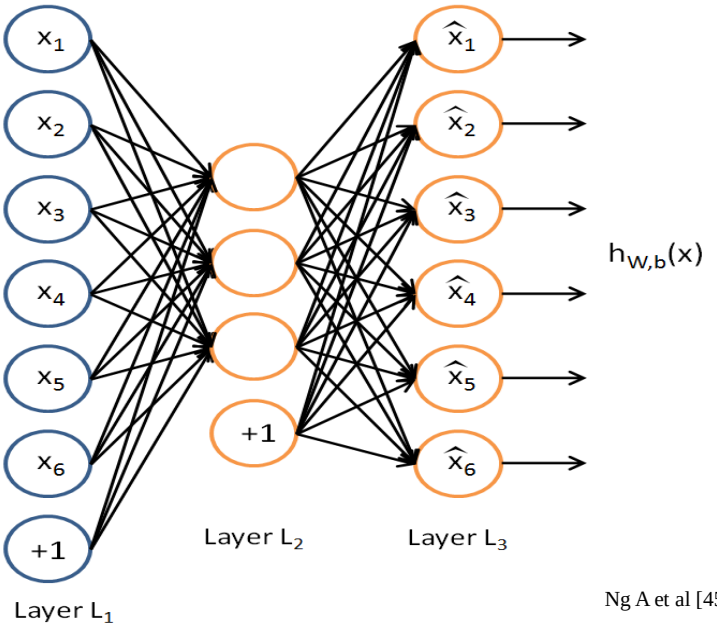


Figura 14: *Estructura d'un autoencoder*

Existeixen diferents variants dels autoencoders, com els Denoising autoencoder, que permeten reduir el soroll de les dades d'entrada, o els Sparse autoencoder, que permet extreure descriptors d'alt nivell de representació de manera no supervisada amb capes ocultes amb més nodes que les capes d'entrada i sortida. Els autoencoders es poden "apilar" de la mateixa manera que les Deep Belief neural networks, rebent el nom de Stacked autoencoders, xarxes més profundes capaces de detectar característiques més complexes.

1.1.4. Justificació del TFM

El coneixement i comprensió dels diferents processos de regulació gènica, entre ells els miRNA, han de permetre explicar les diferències morfològiques i funcionals dels diferents tipus cel·lulars d'un mateix individu, amb un genoma idèntic, i com la desregulació d'aquests mecanismes poden donar lloc a diferents disfuncions i/o malalties. En aquest sentit, els miRNA són un bon marcador de prognosi i tenen un gran potencial terapèutic [46]. És per això que el coneixement dels gens que regula un miRNA concret, per una banda, i el conjunt de miRNA que regulen un gen concret, per l'altra, poden aportar un grau d'informació tant per entendre la diferenciació cel·lular com malalties derivades de la seva desregulació.

Existeixen diferents *softwares* de predicció de dianes de miRNA a l'abast de l'investigador, encara que totes les prediccions *in silico* han de ser corroborades a nivell experimental. Però aquestes prediccions permeten dirigir

la recerca experimental sense que els costos siguin inassumibles. Aquestes eines de predicció s'han fonamentat des dels seus inicis en regles basades en la observació i en la recerca experimental, però encara avui en dia es desconeixen les raons per les quals alguns transcripts són diana d'un miRNA determinat, sobretot amb les dianes no canòniques.

En aquest context, les tècniques de *Machine Learning* poden jugar un paper clau en la recerca de dianes. Aquests algorismes són capaços d'aprendre les característiques que fan que un miRNA reguli un transcript concret i apliquen aquest coneixement a noves parelles de miRNA:mRNA. Actualment existeixen diferents *software* que implementen algorismes de *Machine Learning* com els SVM, autoencoders o xarxes neuronals recurrents, amb uns alts nivells d'encert en les seves prediccions.

Per tot això, s'ha marcat com a objectiu entendre i aplicar diferents algorismes de *Machine Learning* a aquesta problemàtica per dissenyar una eina capaç de fer prediccions suficientment acurades.

1.2. Objectius del Treball

En l'inici del projecte es van marcar els següents objectius:

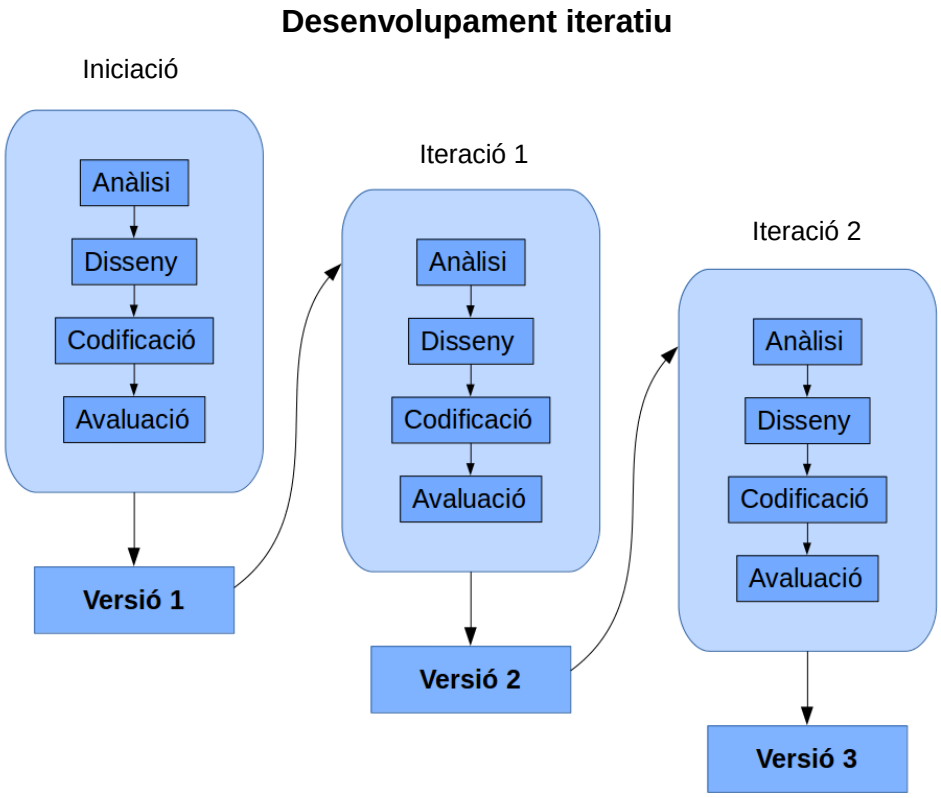
1. Descobrir nous descriptors per a la predicció de mRNA diana.
    1.1. Identificar, mitjançant recerca bibliogràfica, els descriptors més informatius emprats per al modelatge de dianes.
    1.2. Crear nous descriptors a partir d'algorismes de *machine learning* (*Feature learning*).
2. Crear una eina de predicció de dianes miRNA.
    2.3. Avaluar i seleccionar els descriptors més importants per a la predicció de dianes.
    2.4. Avaluar diferents algorismes de *machine learning.*
    2.5. Implementar eina de predicció basada en el millor algorisme segons els resultats del punt anterior.

1.3 Enfocament i mètode seguit

El descobriment de noves dianes de miRNAs es pot enfocar seguint dues estratègies: la recerca experimental o la predicció d'aquestes dianes *in silico.* La primera és una estratègia amb uns elevats costos econòmics i temporals. Gràcies a l'avenç de les capacitats computacionals i a la gran quantitat de

dades que es generen en l'actualitat, la predicció *in silico* de dianes sembla una bona aproximació per generar nou coneixement, tot i que aquestes prediccions cal que siguin validades de forma experimental.

La metodologia ha seguit un desenvolupament iteratiu i creixent, constant d'una etapa d'iniciació, en que s'ha creat una primera versió de l'algorisme, simple i funcional, i una etapa d'iteració, en que l'algorisme s'ha redissenyat en diferents iteracions per obtenir-ne versions millorades.



1.4 Planificació del Treball

El treball final de màster té una durada d'aproximadament 2 mesos (300 hores). Per tal d'assolir els dos objectius marcats anteriorment, s'han proposat una sèrie de tasques a seguir i una planificació temporal d'aquestes per a cada objectiu:

Descobrir nous descriptors (*features*) en la predicció de mRNA diana.

- Conèixer de l'*state of the art* de la predicció de dianes mitjançant la recerca bibliogràfica.
- Conèixer de l'*state of the art* de les tècniques de *feature learning* mitjançant la recerca bibliogràfica.

- Cercar dades amb les que entrenar/testar els descriptors i l'algoritme de predicció.
- Dividir les dades en dos sets: un per descobrir descriptors i l'altre por entrenar l'algorisme de predicció.
- Definir els subsets per a train/test.
- Dissenyar i implementar un algorisme de *feature learning* per obtenir nous descriptors.

Crear una eina de predicció de dianes miRNA.

- Definir els subsets per a train/validation/test.
- Escollir i avaluar el millor set de descriptors.
- Avaluar diferents algorismes de *machine learning* a nivell teòric i escollir el més adequat.
- Implementar un algorisme de *machine learning* de predicció de dianes i comparar-la amb altres eines actuals.

Per a garantir el correcte desenvolupament del TFM, s'han marcat un seguit de dates en que s'ha reportat el desenvolupament del treball. Aquests informes han permès comprovar si algun imprevist ha retardat el pla marcat en la planificació del Pla de treball, o si alguna circumstància ha fet modificar la direcció de la recerca. Les fites marcades (senyalades amb un rombe vermell en el diagrama de Gantt (Figura 15) són les següents:

- 15/03/17 ***PAC1 - Pla de treball.*** Document que té com a finalitat concretar, delimitar i descriure el treball a realitzar, així com marcar els objectius, descriure la metodologia a seguir, i establir una planificació temporal.
- 22/03/17 ***Resum de l'*state of the art.** Breu resum de les diferents eines actuals de predicció de dianes de miRNAs i els diferents predictors emprats. Resum de diferents eines de Feature Learning que permeten el descobriment de nous descriptors a partir de la reducció dimensional de les dades.
- 28/03/17 ***Descripció del dataset.*** Documentar les dades contingudes en el dataset de treball així com la seu origen. Descripció de les diferents etapes de processat abans de ser utilitzades en els diferents algorismes.
- 5/04/17 ***PAC2 - Desenvolupament del treball - Fase 1.*** S'entregarà un primer informe de seguiment del treball. En aquest document s'avaluarà el correcte desenvolupament del TFM.

Caldrà indicar i justificar desviacions temporals i/o canvis en els objectius inicials en cas que n'hi hagués.

- 13/04/17 ***Entrega de l'algorisme de* feature learning.** Es presenta un algorisme de reducció dimensional que processa les dades que han d'entrar a l'algorisme de predicció de dianes. Es descriuen els diferents algorismes testats i s'avalua el seu desenvolupament.

- 24/04/17 ***Conjunt de descriptors seleccionats.*** S'entregarà el set de descriptors que entrenarà l'algorisme i la justificació d'aquesta decisió.

- 1/05/17 ***Comparativa dels diferents algorismes.*** Es presenta un estudi comparatiu de diferents algorismes i de la seva aplicació en el problema objecte d'aquest treball. S'escull l'algorisme utilitzat en l'eina de predicció i es valora a partir de diferents estadístics.

- 10/05/17 ***PAC3 - Desenvolupament del treball - Fase 2.*** S'entrega un segon informe de seguiment del treball. Com en el cas anterior, el document serveix per a avaluar el correcte desenvolupament del TFM, així com per detectar desviacions temporals i/o canvis en els objectius inicials, en cas que n'hi hagi hagut.

- 16/05/17 ***Implementació de l'algorisme de predicció.*** Un cop tenim els descriptors i l'algorisme, s'implementa l'eina de predicció de dianes de miRNA, i es compara amb altres eines actuals de predicció *in silico.*

- 24/05/17 ***Memòria i presentació del treball final.*** La memòria del TFM és el document final en que es descriu de forma detallada i ampliada els objectius finals, la metodologia seguida i els resultats obtinguts. També es fa entrega d'una presentació en la que es resumeix de forma clara i concisa el treball realitzat al llarg del semestre i els resultats obtinguts, oferint una perspectiva general del TFM i recollint-ne els aspectes més rellevants.

- 07/06/17 – 21/06/17 ***Defensa pública - Tribunal TFM.*** Presentació i defensa del treball davant d'un tribunal, qui podrà formular les preguntes que consideri oportunes i que finalment emetrà una avaluació final del treball.

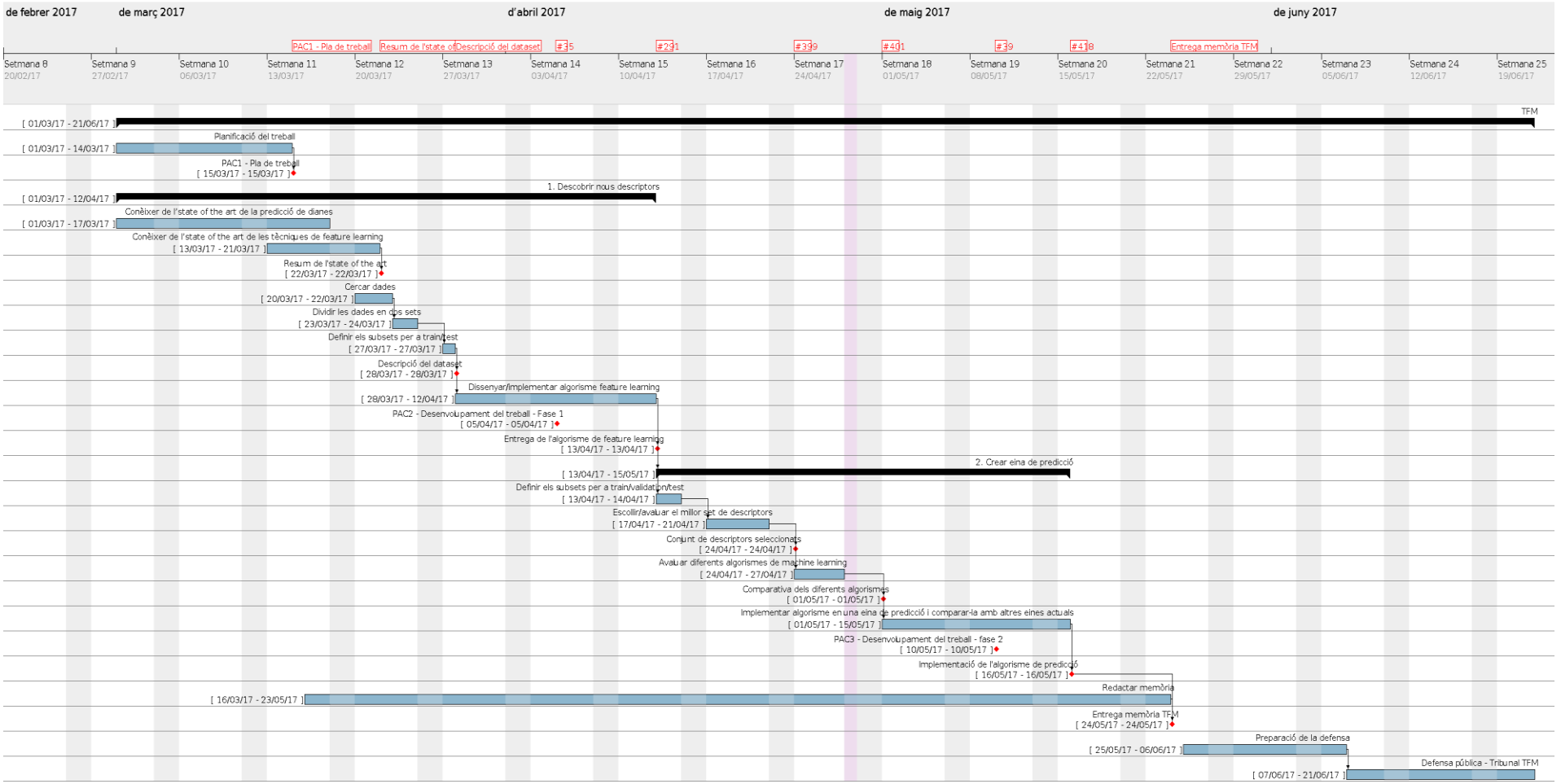


Figura 15: *Diagrama de Gantt amb la planificació temporal seguida durant el desenvolupament del projecte. Les diferents fites es representen amb un rombe vermell*

1.5 Breu sumari de productes obtinguts

L'objectiu final del treball és l'elaboració d'una eina de predicció de dianes de miRNA. Aquesta eina, anomenada miRNAforest, permet introduir la seqüència d'un miRNA conegut i la seqüència del 3'UTR d'un transcript sospitós de ser regulat per aquest miRNA. A través d'un model generat a partir de tècniques de *Machine Learning,* l'usuari obté una predicció de si el transcript en qüestió és una potencial diana del miRNA, o per el contrari, la traducció del transcript no es veu afectada directament pel miRNA objecte d'estudi.

1.6. Breu descripció dels altres capítols de la memòria

2. Materials i mètodes: Descripció del conjunt de dades utilitzat en aquest TFM, la seva procedència i el processat aplicat per tal que siguin útils per a generar el model de predicció.

3. Descobriment de nous descriptors: Descripció de l'algorisme triat per a la reducció dimensional de les dades i les diferents etapes del procés d'aprenentatge.

4. Algorisme de predicció: Descripció de l'algorisme de predicció de dianes de miRNA i de les diferents iteracions fins a assolir el model final. Avaluació del model definitiu.

3. Conclusions: Anàlisi crític del grau d'assoliment dels objectius i del desenvolupament del projecte. Reflexió de possibles línies de treball futures.

4. Glossari: Definició dels termes i acrònims més rellevants utilitzats en la Memòria.

5. Bibliografia: Llista numerada de referències bibliogràfiques.

6. Annexes: Diferents scripts rellevants dins del projecte.

2. Materials i mètodes

En el camp del Machine Learning, s'anomenen materials al conjunt de dades que permetrà generar un model. En aquesta secció es descriuen l'origen de les dades utilitzades en aquest TFM, així com les diferents manipulacions per a obtenir un conjunt de descriptors útil per a assolir els objectius marcats.

2.1. Conjunt de dades

Aquest treball té la intenció de generar un model de predicció de dianes de miRNA basat en algorismes de *Machine learning.* En aquest sentit, és fonamental tenir un bon set de dades que permeti generar un model robust que obtingui unes prediccions fiables. Aquestes dades han d'haver estat validades experimentalment, no podem generar un model de predicció basat en prediccions obtingudes per altres eines. Existeixen diferents repositoris a la xarxa on es poden trobar llistes validades de parelles de miRNA:mRNA, uns enfocats a l'estudi d'una única espècie, d'altres centrant-se en la relació de miRNA amb alguna malaltia o grup de malalties. Alguns dels repositoris més importants en xarxa són miRecords (http://c1.accurascience.com/miRecords/) [47], miRTarbase [48] (http://mirtarbase.mbc.nctu.edu.tw/), TarBase 7.0 [49] (http://diana.imis.athena-innovation.gr/DianaTools/index.php?r=tarbase/index) i miRBase [50] (http://www.mirbase.org/). Com descriu Yun Ji Lee et al. [51], cada repositori té els seus avantatges i desavantatge, però un dels problemes més importants en la majoria d'ells és la manca (o baix nombre) de dades de mRNA descartats com a diana, és a dir, d'exemples negatius validats experimentalment. Cal tenir en compte que per tal d'aplicar un algorisme de *Machine learning* són necessaris tant exemples positius com negatius, i és per això que sovint es generen seqüències aleatòries que serveixin d'exemples negatius (mock miRNA) [52]. Aquesta aproximació, tot i ser utilitzada en alguns algorismes de predicció [24], corre el perill que l'algorisme de *machine learning* aprengui a classificar entre miRNAs reals i mock miRNAs, de manera que no valori la seqüència 3'UTR del transcript potencial diana, impedint la possibilitat d'aprendre nous descriptors vàlids per a la seva classificació.

Un altre dels problemes que tenen la majoria de repositoris és que les llistes són de parelles miRNA:gen, i no miRNA:transcript. Les dianes dels miRNA es troben en els 3'UTR dels mRNA, és a dir, són específiques de transcript. Cal tenir en compte que cada gen pot donar lloc a més d'un transcripts per *splicing*, diferint en mida i seqüència. És important tenir doncs el transcript que ha estat validat per tal d'obtenir la seqüència del 3'UTR correcte.

Tenint en compte totes aquestes consideracions, el dataset escollit per a la realització d'aquest TFM són els exemples positius utilitzats per Mark Menor et al. [53], que es descriuran a continuació, i els exemples negatius validats experimentalment del repositori TarBase.

Degut a les limitacions de temps, aquest treball s'ha centrat només en dades de miRNAs humans, tot i que la mateixa metodologia podria ser aplicada a altres espècies. Les dades obtingudes de Mark Menor et al. contenen exemples positius que provenen dels repositoris miRecords [54] i miRTarBase [55]. Del primer repositori (v19) s'han utilitzat totes les parelles de miRNA i gens humans, i del segon, les parelles amb fortes evidències experimentals. L'arxiu descarregat consta de dues columnes, una amb l'identificador del miRNA i l'altre amb l'identificador RefSeq del transcript, i 2937 observacions. Les seqüències dels miRNAs madurs s'han descarregat de miRBase (Release 19), i les seqüències dels transcripts s'han obtingut amb el paquet *biomaRt* d'R.

L'arxiu descarregat del repositori TarBase consta de quatre columnes, corresponents a l'identificador de miRNA, el nom del gen diana, l'identificador ensembl del gen i una variable binaria, 0 per als exemples negatius, i 1082 observacions. Igual que amb els casos positius, les seqüències dels transcripts s'han obtingut amb el paquet *biomaRt* d'R. TarBase testeja diversos 3'UTR de cada gen, tot i que finalment només dona el nom i la ID del gen testat.

En la taula 2 es resumeix el dataset un cop processades les dades, havent buscat les seqüències tant dels miRNA com dels 3'UTR dels transcripts diana i eliminat els exemples dels que no s'han pogut obtenir la seqüència del 3'UTR.

	Nº de miRNA	**Nº de gens**	**Nº de transcripts**	**Nº parelles miRNA:mRNA**
Dades positives	287	1485	1525	2711
Dades negatives	141	729	2757	3989
Dataset	322	2003	4033	6700

Taula 2: *Sumari del dataset emprat*

Aquest dataset es pot consultar al material suplementari en format .csv. El fitxer conté les següents columnes:

- ***target:*** Variable binaria, 0 en els casos de dianes negatives, 1 en els exemples positius
- ***miR_ID:*** Identificador del miRNA
- ***miRNA_seq:*** Seqüència de nucleòtids del miRNA madur
- ***gene_symbol:*** Nom del gen diana

- ***mRNA_ID:*** Identificador del mRNA
- ***3utr:*** Seqüència del 3'UTR del mRNA diana

2.2. Seqüència diana

El dataset amb el que es treballarà, com ja s'ha definit, consta de 6700 exemples i 6 variables. L'objectiu del projecte és entrenar un algorisme que, amb les seqüències del miRNA i l'extrem 3'UTR del mRNA, sigui capaç de predir si es tracta d'una possible diana o no. En primer lloc cal delimitar el fragment de la seqüència del 3'UTR que entrarem a l'algorisme, ja que aquestes són de longitud variable, poden anar de les poques bases a més de 30.000. Les dades utilitzades no defineixen la seqüència real del 3'UTR del mRNA a on el miRNA s'uneix, de manera que s'ha hagut de buscar possibles punts d'unió. Per facilitat, i tenint en compte les limitacions temporals, per cada parella de miRNA:mRNA s'ha buscat una de les seqüències llavor més utilitzades en els diferents softwares de predicció actuals [56-59]:

- 8mer: aparellament perfecte WC dels nt 2-8 de la llavor i una A en posició 1.
- 7mer-A1: aparellament perfecte WC dels nt 2-7 de la llavor i una A en posició 1.
- 7mer-m8: aparellament perfecte WC dels nt 2-8 de la llavor.
- 6mer: aparellament perfecte WC (Watson-Crick) dels nt 2-7 de la llavor.

Dins la seqüència del 3'UTR de cada parella, s'ha buscat en aquest ordre una seqüència complementària a la llavor del miRNA, de manera que si en una parella concreta s'ha trobat una llavor 8mer s'ha escollit aquesta com a possible diana, i en cas contrari s'ha buscat una seqüència llavor 7mer-A1 i així successivament. En cas de no trobar cap de les seqüències llavor anteriors, s'han buscat aquestes seqüències, però permetent aparellaments GU. S'ha escollit la diana amb menys aparellaments GU. Tant el tipus de llavor com el nombre d'aparellaments GU s'han guardat com a variables per comprovar si poden millorar l'algorisme de classificació.

La seqüència del miRNA es divideix en tres regions, la regió 5 (seqüència llavor, $z_1 – z_8$) , la regió 3 ($z_9 – z_{20}$) i la regió total ($z_1 – z_{20}$) (Figura 16). S'ha determinat una regió de 30 bases del 3'UTR, des de la base complementaria a la posició z_1 i en sentit 5', que seran les que entrenaran l'algorisme. S'han descartat les parelles de miRNA:mRNA en les que no s'ha trobat cap de les

llavors anteriors. Després d'aquest processat, ens queda un dataset amb 4950 exemples, 2504 de dianes negatives i 2446 exemples de dianes positives.

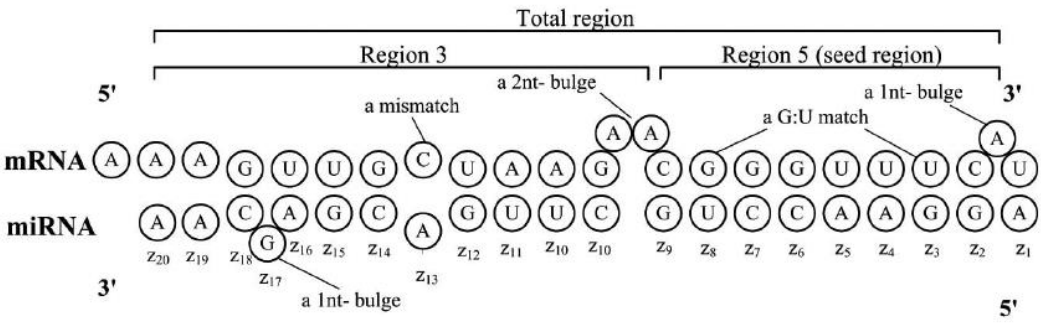


Figura 16: *Regions del miRNA*

2.3. Codificació One-Hot

Per entrenar els algorismes de *machine learning* és necessari que les variables categòriques es transformin en variables numèriques. Per això, es divideix cada seqüència (tant miRNA com mRNA) en els diferents nucleòtids que la componen, i a cadascun d'ells se li assigna una codificació numèrica. En aquest cas s'ha optat per una codificació one-hot, que consisteix en assignar un codi de n bits, en que un únic d'ells té un valor d'1 i la resta són 0 [61]. El nombre de bits correspon al nombre de nivells que té la variable categòrica, en el nostre cas, cada posició constarà de 4 bits (tenim 4 nivells, un per cada possible nucleòtid). Per tant, cada posició de la seqüència es descompondrà en 4 noves variables. La codificació que s'ha seguit és la següent:

{A = (1,0,0,0), U = (0,1,0,0), C = (0,0,1,0), G = (0,0,0,1), N = (0,0,0,0)}

Per tal que totes les seqüències de miRNA tinguin el mateix nombre de variables, s'han ajustat totes fins a 25 nucleòtids, ja que alguns dels miRNA tenen aquesta allargada. Als miRNA que tinguin menys nucleòtids se'ls afegeixen tantes Ns com nucleòtid absents, que codificaran com a (0,0,0,0). Les 100 variables que corresponen al miRNA (25 nucleòtids x 4) i les 120 (30 nucleòtids x 4) del mRNA s'uneixen, de manera que cada parella de miRNA:mRNA queda representada per 220 variables binàries.

2.4. Divisió del dataset

Tenim un dataset amb 4950 exemples, 2504 de dianes negatives i 2446 exemples de dianes positives. S'ha dividit en dos subsets, un per assolir el primer objectiu, descobrir nous descriptors, i l'altre per a generar un model de predicció. Per tal d'assegurar-nos tenir dades representatives de les dues classes, s'han dividir de forma aleatòria les dades positives i negatives en dos, de manera que s'han obtingut 2 subsets de dades amb 2475 exemples, 1252 de positius i 1223 de negatius, cadascun. Finalment, cada subset s'ha dividit en dos grups, un grup d'aprenentatge i l'altre d'avaluació, amb el 75% i el 25% de les dades respectivament. Totes les divisions s'han set de forma aleatòria i mantenint la proporció d'observacions positives i negatives.

3. Descobriment de nous descriptors

Per optimitzar l'algorisme de classificació, s'ha aplicat un algorisme de reducció dimensional. Existeixen diferents mètodes estadístics, com el PCA i l'ICA, i tècniques de *machine learning*. S'ha optat per aplicar un autoencoder, una xarxa neuronal capaç de detectar els descriptors més rellevants. Els autoencoders prenen els descriptors de les nostres dades com a entrada, i aquests mateixos com a sortida, de manera que en la o les capes ocultes es comprimeixen les dades per a posteriorment reproduir-les novament. És precisament una capa oculta, amb menys descriptors dels que inicialment s'han entrat a l'algorisme, la que s'utilitzarà per a entrenar l'algorisme de classificació. S'han avaluat diferents paràmetres, que es descriuran posteriorment, per tal d'obtenir els millors resultats possibles en la reconstrucció de les variables originals.

L'algorisme s'ha construït amb el paquet keras de python [62]. S'han estudiat diferents arquitectures d'autoencoder amb una sola capa oculta, amb diferents nombre de nodes i funcions d'activació. Les limitacions de temps han impedit explorar a fons altres configuracions, però les dades obtingudes amb algunes estructures d'stacked autoencoder fan pensar que no milloren els resultats obtinguts amb autoencoders d'una sola capa oculta. Tenint en compte que el nombre de combinacions és infinita i el temps limitat, s'han donat com a bons els resultats obtinguts.

L'algorisme *Autoencoder* requereix definir diferents paràmetre, tot i que el paquet keras permet utilitzar alguns valors per defecte. Els següents són paràmetres que s'han afinat durant l'entrenament i han tingut un efecte positiu a l'hora de reconstruir les variables d'entrada:

> ***Funció d'activació:*** En una ANN, cada node (o neurona) rep uns valors d'entrada, als quals se'ls aplica una funció, anomenada funció d'activació, per donar una senyal de sortida, que serviran com a capa visible de la següent capa de la xarxa [63]. Existeixen diferents funcions d'activació utilitzades en les ANN. En la taula 3 es descriuen les funcions que s'han avaluat en l'autoencoder, suportades pel paquet keras.
>
> ***Funció cost:*** Amb l'objectiu de trobar una solució òptima, durant l'etapa d'aprenentatge cal reduir el valor de la funció cost en el conjunt de dades 'test' [64]. Aquesta funció calcula la distància (o desviació) de la solució obtinguda en una època respecte a la solució òptima. A cada època de l'etapa d'aprenentatge, es calcula el cost del model obtingut, fins a obtenir

la millor solució possible, és a dir, la que obté unes prediccions amb menys desviació dels valors reals. Existeixen diferents funcions de cost, incloses al paquet keras, com la funció hinge ($\ell(y) = \max(0, 1 - t \cdot y)$) o la Cross entropy ($V(f(\vec{x}), t) = -t \ln(f(\vec{x})) - (1 - t) \ln(1 - f(\vec{x}))$). S'han realitzat algunes proves amb les diferents funcions suportades pel paquet keras, optant finalment per la funció error quadràtic mig ($\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2$) ja que és la que ha obtingut uns millors resultats en la reconstrucció de les dades originals.

Funció	Equació
elu (Exponential linear unit)	$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$
ReLU (Rectified linear unit)	$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$
softmax	$f(\vec{x})_i = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}} \text{ for i} = 1, \ldots, \text{K}$
softplus	$f(x) = \ln(1 + e^x)$
softsign	$f(x) = \frac{x}{1 + \lvert x \rvert}$
tanh	$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
sigmoid	$f(x) = \frac{1}{1 + e^{-x}}$
hard sigmoid	$f(x) = clip\left(\frac{x + 1}{2}, 0, 1\right)$
linear	$f(x) = x$

Taula 3: *Algunes funcions d'activació utilitzades en ANN*

Optimitzador: El descens de gradient és un mètode general minimitzador per a qualsevol funció [65]. Aplicat a una xarxa neuronal, s'encarrega de buscar de forma iterativa en l'espai de paràmetres el conjunt que minimitza la funció cost, objectiu a l'hora d'entrenar un model. Es comença amb la xarxa neuronal amb un vector de paràmetres inicial que cal anar millorant a cada iteració (època), de manera que la funció cost es vagi reduint (figura 17). El paquet keras permet aplicar diferents

optimitzadors, d'entre els que s'ha escollit Nesterov Adam optimizer després de comprovar el desenvolupament dels diferents mètodes que permet el paquet keras.

Nombre d'èpoques: Una època és un cicle d'aprenentatge, cada iteració, en que tots els exemples del set de dades han passat a través de la xarxa neuronal. En finalitzar una època, la xarxa s'adapta als resultats obtinguts per tal de disminuir el cost. S'ha establert el nombre màxim d'èpoques en 1000 per a l'entrenament de l'autoencoder, aturant el procés si no es millora el valor del cost en 10 èpoques consecutives.

Regularitzador: La regularització penalitza la complexitat del model, de manera que en ser incorporat a la funció de cost, permet reduir l'overfitting, evitant que un paràmetre tingui més pes que el conjunt de la resta [66]. S'ha escollit la regularització L2 (o descomposició de pesos), que afavoreix que la suma del quadrats dels paràmetres de la xarxa sigui petita.

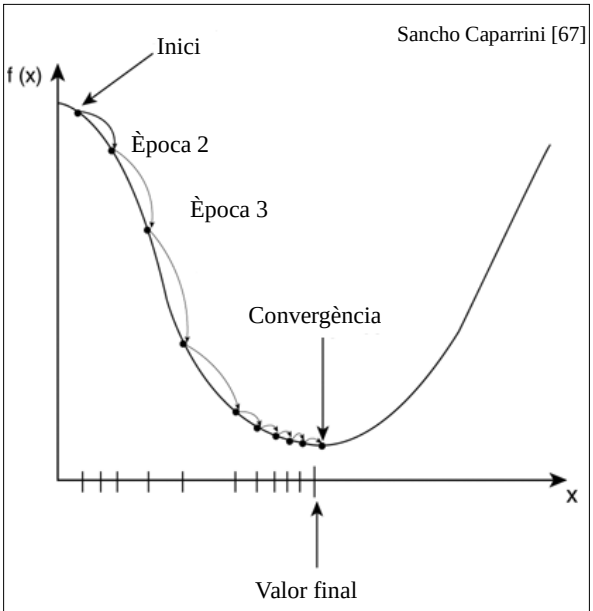


Figura 17: *Optimització de la funció. A cada època es redueix la funció cos fins a arribar a una solució òptima (mínim global) o subòptima (mínim local)*

Els *autoencoders* consten de dues etapes, l'*encoder,* en que hi ha una reducció dimensional de les dades d'entrada, i el *decoder,* en que es reconstrueixen els valor inicials a partir de les dades comprimides per l'*encoder*. Amb els paràmetres anteriors, s'han dissenyat tres arquitectures de d'autoencoders diferents. Aquestes ANN consten de tres capes, la primera capa visible conté els nodes d'entrada, en aquest cas 220, corresponents a la codificació one-hot de les seqüències del miRNA i la regió diana del 3'UTR del transcript a testar,

una capa oculta, amb un nombre de nodes inferior a la capa visible, i una capa de sortida amb el mateix nombre de nodes que la capa d'entrada, per tant, 220 nodes. S'han testat diferents dimensions a la capa oculta (20, 40 i 60 nodes) per comprovar el grau de reconstrucció per part del *decoder* de les dades comprimides per l'*encoder.* Els resultats es mostren a la taula 4.

Per altra banda, s'ha experimentat amb un *stacked autoencoder,* que no és més que un *autoencoder* multicapa. L'estructura de l'*stacked autoencoder* consta de 5 capes, les capes d'entrada i de sortida de 220 nodes, i tres capes ocultes, amb 130, 60 i 130 nodes respectivament. Es fa difícil definir la millor estructura dels *stacked autoencoder* ja que el nombre de combinacions és infinit. S'ha volgut mantenir el nombre de nodes de la capa central, la que servirà per a alimentar l'algorisme de predicció. S'ha mantingut una estructura simètrica i s'ha experimentat amb diferents valors per a les capes 2 i 4. Finalment, s'ha entrenat una xarxa amb una estructura 220-130-60-130-220. Els resultats obtinguts s'han incorporat a la taula 4.

Algorisme	Arquitectura	Funció d'activació	Exactitud	Sensitivitat	Especificitat	AUC	nº èpoques
Autoencoder	220-20-220	elu	0.8783	0.9677	0.5920	0.9008	260
	220-40-220	linear	0.923	0.9656	0.7864	0.9547	123
	220-60-220	softplus	0.9673	0.9838	0.9146	0.986	248
Stacked autoencoder	220-130-60-130-220	linear	0.9507	0.9736	0.8774	0.9801	123

Taula 4: *Sumari de les diferents arquitectures d'autoencoders avaluades*

Com es pot observar en la taula 4, l'estructura d'autoencoder amb un millor grau de reconstrucció és aquella amb 60 nodes a la capa oculta i amb la funció softplus com a funció d'activació. Els autoencoders d'una sola capa han obtingut millors resultats en la reconstrucció de les dades originals que els *stacked autoencoder,* de manera que no s'han explorat a fons configuracions alternatives. En les figures 18 i 19 podem veure l'evolució de la funció de cost durant l'entrenament dels diferents autoencoders i la representació de les corbes ROC dels algorismes de la taula 4, respectivament.

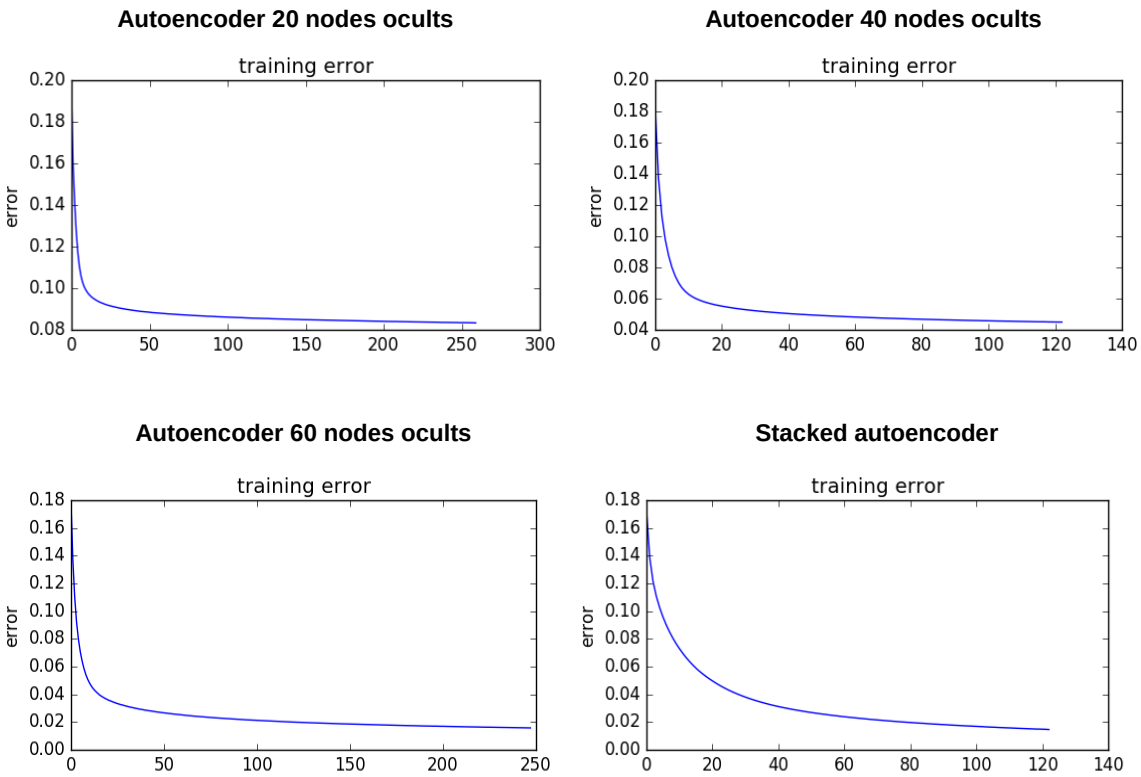


Figura 18: *Evolució de l'error en cada època per els diferents autoencoders testats*

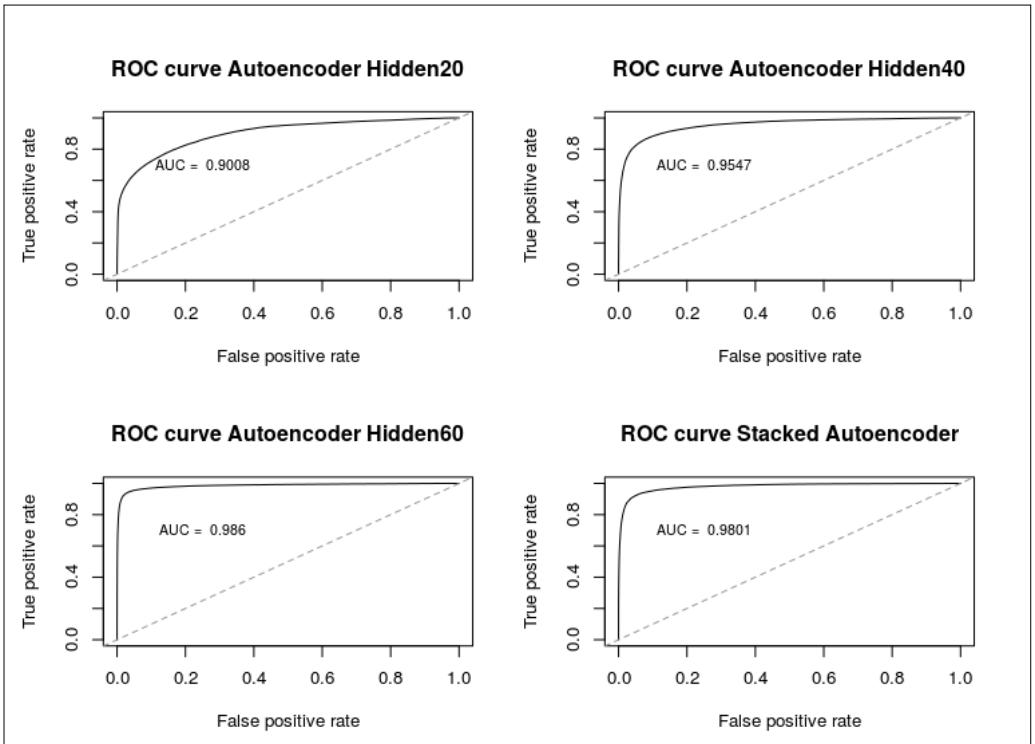


Figura 19: *Corbes ROC i rea sota la corba (AUC) del diferents Autoencoders testats*

4. Algorisme de predicció

L'objectiu principal d'aquest treball és generar un algorisme que permeti identificar possibles dianes de miRNA. El primer pas ha estat generar nous descriptors mitjançant la reducció dimensional a partir de les seqüències del miRNA i de la potencial diana a avaluar gràcies a un autoencoder. S'han avaluat diferents arquitectures, de manera que s'ha passat de 220 dimensions a 60 (Hidden60), 40 (Hidden40) i 20 (Hidden20). A més d'aquests sets de descriptors, també s'han generat descriptors referents al tipus de llavor trobada (8mer, 7mer-A1, 7mer-m8 o 6mer) i el nombre d'aparellaments GU entre la seqüència llavor del miRNA i la seqüència del 3'UTR considerada com a possible diana del miRNA en qüestió.

En aquest punt, s'han considerat diferents algorismes de Machine Learning per classificar mRNAs com a dianes o no dianes de miRNA. Cada algorisme de predicció s'ha entrenat amb 8 possibles sets de descriptors, per una banda, els sets de descriptors obtinguts amb les diferents estructures d'autoencoders descrits a l'apartat 3 (Descobriment de nous descriptors), és a dir, els obtinguts amb Hidden60 (autoencoder amb 60 nodes ocults), els obtinguts amb Hidden40 i els obtinguts amb Hidden20, a més de les dades sense ser processades amb cap autoencoder, els 220 descriptors obtinguts amb la codificació one-hot de l'apartat 2.3 (Codificació One-Hot). Finalment, cadascun d'ells s'ha complementat amb les variables referents al tipus de llavor detectada i el nombre d'aparellaments G:U, descrits a l'apartat 2.2 (Seqüència diana).

El set encarregat d'entrenar l'algorisme conté 1252 exemples negatius i 1223 exemples positius, que s'han dividit en un grup 'train' (75%) i un grup 'test' (25%). Tots els algorismes s'han entrenat amb el mètode 10 fold cross-validation, que permet estimar l'exactitud de l'algorisme dividint el set 'train' en deu parts, realitzar deu cicles d'aprenentatge utilitzant un dels deu subsets per a validar l'algorisme. D'aquesta manera es pot afinar l'algorisme sense utilitzar el set 'test'. Cal tenir en compte que el set 'test' no ha d'intervenir en cap dels processos d'aprenentatge, per tant, no ens podem basar en l'exactitud obtinguda amb el set test per afinar els diferents paràmetres. Un cop s'han afinat els diferents algorismes, s'escollirà el millor segons els resultats de la validació i es valorarà el seu desenvolupament amb el subset 'test'.

S'han provat diferents configuracions de paràmetres en cada algorisme per tal de millorar-ne l'exactitud, en la figura 20 es pot veure el l'exactitud estimada dels diferent algorismes segons les dades d'entrada, i en la taula 4 es resumeixen els diferents algorismes i l'exactitud estimada gràcies a la validació creuada (10-fold cross-validation).

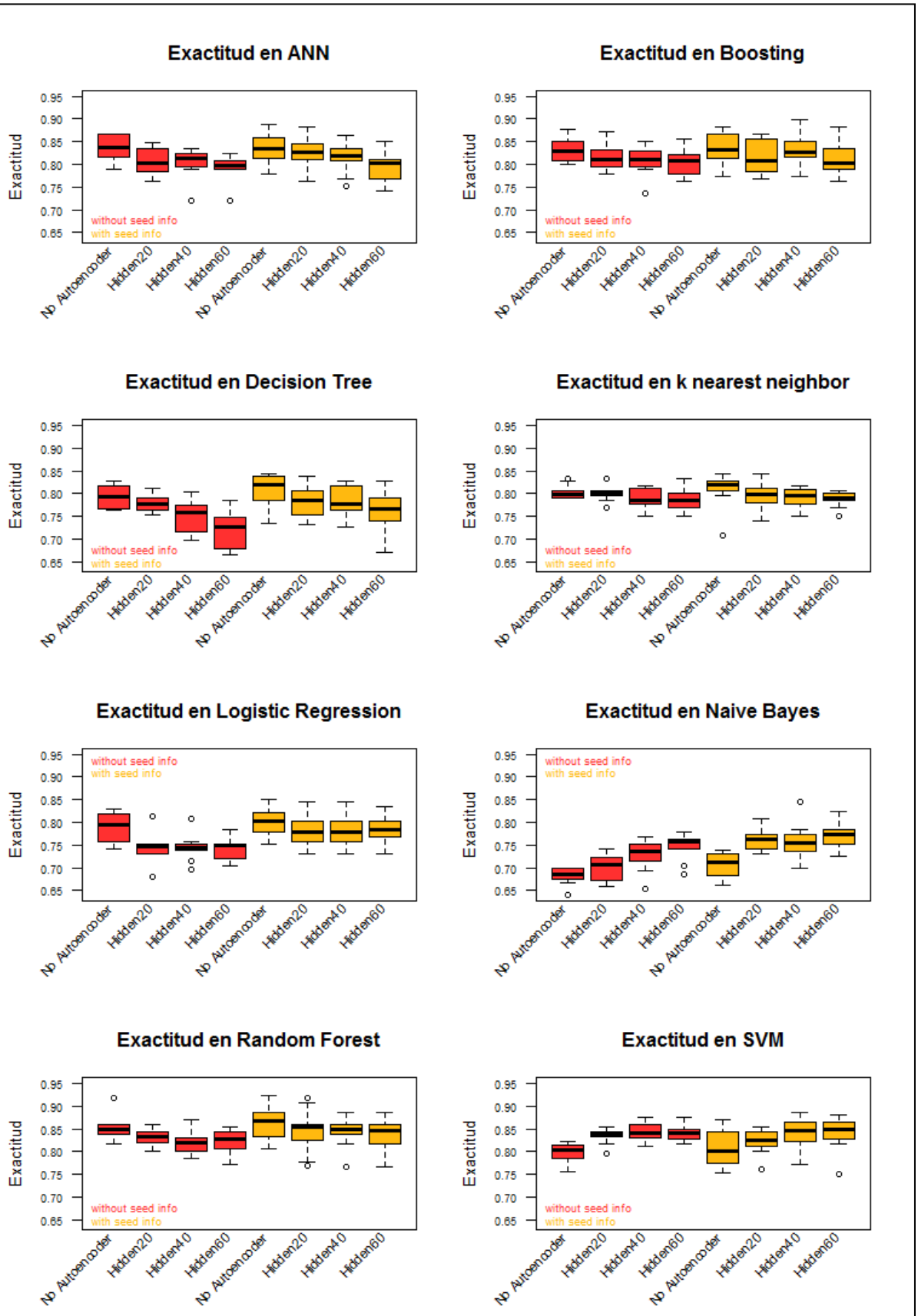


***Figura 20**: Estimació de l'exactitud de la predicció dels diferents algorismes segons els descriptors d'entrada*

Algorisme	Seed info	Encoder	exactitud estimada	Desviació estàndard
Naive Bayes	SI	Hidden60	0.7683	0.0292
kNN	SI	NO	0.8092	0.0383
Logistic Regression	SI	NO	0.8022	0.0304
Decision Tree	SI	NO	0.8065	0.0376
Random Forest	SI	NO	0.8648	0.0364
SVM	NO	Hidden40	0.8437	0.0193
Boosting	NO	NO	0.8329	0.0277
ANN	SI	NO	0.8335	0.0347

Taula 5: *Sumari dels diferents algorismes testats, amb la mitjana de l'exactitud obtinguda per cross-validation i la seva desviació estàndard. Es mostra per cada algorisme el millor resultat d'entre les diferents combinacions de dades d'entrada i paràmetres testats.*

Com es pot comprovar en la taula 5, el nombre de descriptors varia entre els diferents algorismes, des dels que tenen un millor desenvolupament amb els descriptors derivats dels diferents autoencoders, als que tenen una millor exactitud amb els 220 descriptors derivats de la codificació one-hot, tot i que predomina l'entrada de les 220 variables de la codificació one-hot. També es pot observar que la informació referent al tipus de llavor millora gairebé tots els algorismes. L'algorisme amb una millor exactitud estimada es el Random Forest, amb una exactitud estimada del 86,48% +/- 2,05. L'algorisme amb millor exactitud que utilitza els predictors procedents d'un autoencoder és el SVM, amb una exactitud estimada del 84,37% +/- 1,93. En la figura 21 es comparen els diferents algorismes.

Amb aquests resultats, s'ha utilitzat l'algorisme *Random Forest* per a l'eina de predicció. Finalment, amb el subset 'test' es valora el model. Els resultats obtinguts es troben en la següent taula de contingència:

		Real		
		No diana	**Diana**	
Predicció	**No diana**	265 TN	40 FN	305
	Diana	48 FP	266 TP	314
		313	306	619

Exactitud:	0.8578
Sensitivitat:	0.8693
Especificitat:	0.8466
Precisió:	0.8471
F1 score:	0.8581

TP: verdader positiu
TN: verdader negatiu
FP: fals positiu
FN: fals negatiu

Taula 6: Taula de contingència de les prediccions del model de predicció de dianes

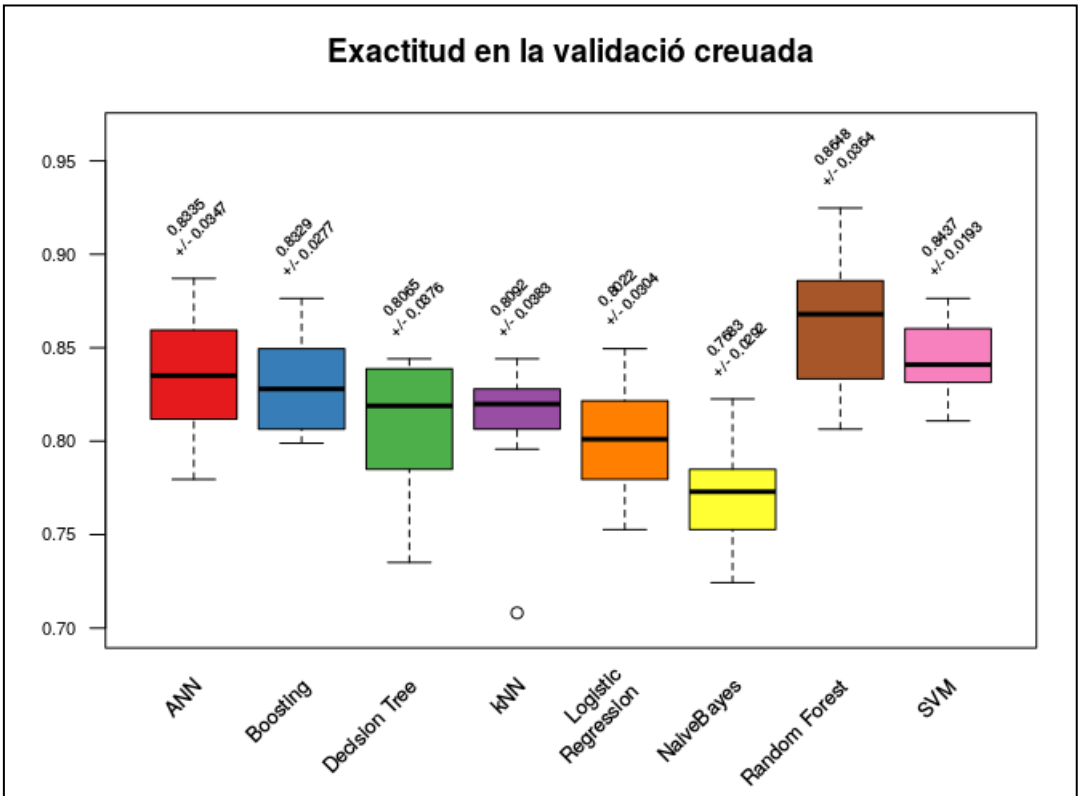


Figura 21: Estimació de l'exactitud del millor model de cada algorisme. *Es mostra per cada algorisme el millor resultat d'entre les diferents combinacions de dades d'entrada i paràmetres testats*

Els diferents estadístics calculats són:

- Exactitud: És la proporció de resultats correctament classificats d'entre totes les prediccions:

$$exactitud = \frac{TP + TN}{TP + FP + FN + TN}$$

- Sensitivitat: És la capacitat del model a predir com a positiva les verdaderes dianes, és a dir, és la capacitat del model de detectar dianes:

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

- Especificitat: És la capacitat del model a predir com a negativa una presumpta diana que realment no ho és, és a dir, és la capacitat del model de descartar els casos negatius:

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

- Precisió: És la probabilitat que, havent estat classificada com a diana, aquesta ho sigui realment:

$$Precisió = \frac{TP}{FP + TP}$$

- F1-score: És la mitjana harmònica que combina els valors de precisió i sensitivitat:

$$F_1 = 2 \cdot \frac{Precisió \cdot Sensitivitat}{Precisió + Sensitivitat} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

5. Conclusions

En l'inici d'aquest TFM es van marcar dos objectius, el descobriment de nous descriptors per a la predicció de potencials diana de miRNA gràcies a tècniques de Feature Learning, i el disseny d'una eina de predicció de dianes de miRNA basada en models generats a partir d'algorismes de Machine Learning.

El primer objectiu no s'ha assolit completament. S'han utilitzat diferents arquitectures d'autoencoder per tal de reduir la dimensionalitat. Els resultats obtinguts en la reconstrucció de les dades d'entrada han estat positius en l'autoencoder amb 60 nodes ocults, que té un nivell d'exactitud del 96.73 %, i força bons en l'autoencoder amb 40 nodes ocults, amb una exactitud del 92.3%. Tanmateix, aquests nous descriptors no són fàcilment interpretables. D'altra banda, aquests descriptors s'han utilitzat per generar el model de predicció. L'exactitud estimada d'un model de SVM entrenat amb els descriptors obtinguts de l'autoencoder de 40 nodes ocults és del 84.37 %, un bon nivell d'encert si tenim en compte que el millor model obtingut en aquest treball té una exactitud estimada del 86.48 %. És possible que si les dianes no canòniques haguessin format part dels datasets de 'train' i 'test', els autoencoders sí haguessin millorat la qualitat de la classificació. També cal tenir en compte que altres mètodes de reducció dimensional, tals com PCA o ICA, haurien donat bons resultats, però les restriccions en els terminis d'entrega han limitat el nombre d'iteracions en aquesta (i també en les posteriors) fases del projecte.

Pel que fa al segon objectiu del treball, els resultats són satisfactoris. S'han testat 8 algorismes de predicció diferents (Naive Bayes, k-Nearest Neighbour, Logistic Regression, Decision Tree, Random forest, Support vector machine, Boosting i Xarxes Neuronals) amb diferents sets de paràmetres i de dades d'entrada. Tots els models finals tenen una exactitud estimada superior al 80 %, a excepció de Naive Bayes, un algorisme que pressuposa la independència de les variables, fet que no es compleix en aquest cas. L'algorisme de predicció ha estat implementat utilitzant el model de Random Forest, obtenint una exactitud de 0.8578, una sensitivitat del 0.8693, especificitat de 0.8471 i una puntuació F1 de 0.8581. Aquests valors són força bons comparats amb altres softwares de predicció actuals. A la taula 7 es resumeix l'avaluació feta per Lee et al. [24] de diferents eines, utilitzant les dades positives de Menor et al. i generant mock miRNA com a dades negatives. Tot i que la comparació no és exacte, ja que Lee et al. no utilitzen les mateixes dades per a avaluar els diferents softwares que les dades que s'han utilitzat per valorar el model descrit en aquest treball,

si que permet fer una comparació aproximada. L'algorisme proposat només es veu superat per deepTarget en exactitud i F1-score.

	MBSTAR	miRanda	PITA	RNA22	TargetScan	TargetSpy	deepTarget
accuracy	0.5805	0.7737	0.5870	0.5820	0.7125	0.6364	**0.9641**
sensitivity	0.5308	0.6122	0.9301	0.4006	0.7980	0.4934	**0.9389**
specificity	0.5308	0.9228	0.2703	0.7494	0.6336	0.7657	**0.9706**
F-measure	0.5921	0.7220	0.6837	0.4791	0.7271	0.5672	**0.9105**
PPV	0.5551	0.8797	0.5405	0.5960	0.6677	0.6616	**0.8848**
NPV	0.6114	0.7206	0.8074	0.5753	0.7727	0.6223	**0.9845**

Taula 7: *Comparativa de diferents softwares de predicció de dianes, extreta de Lee et al. [24]*

Un punt clau i que va dificultar els inicis del projecte va ser l'obtenció de dades. En primer lloc, i degut a les limitacions temporals, s'ha limitat el treball a la cerca de dianes de miRNA humans, tot i que la metodologia emprada és aplicable a d'altres espècies. Aquesta elecció no és trivial, existeixen més dades de miRNA humans i les seves dianes que de cap altra espècie. Tot i això, els diferents repositòris presenten diferents mancances. La majoria de repositoris de miRNA no presenten la seqüència concreta del mRNA a la que s'uneix el miRNA, si no que són llistes de miRNA:gen, o en el millor dels casos, miRNA:transcript. Per això s'han hagut de seleccionar les seqüències dins el 3'UTR que, complint una sèrie de regles, siguin les més versemblants. Una altra dificultat és la que s'ha esmentat anteriorment, la manca de concreció de quin o quins de tots els possibles transcripts que codifica un gen concret és el que es veu regulat per un miRNA donat. Mitjançant *splicing*, un gen pot donar lloc a diferents transcripts, i aquests poden tenir un 3'UTR diferent en mida i seqüència de nucleòtids, de manera que una diana pot no estar present en tots els transcripts del gen. És per tot això que un nombre més o menys elevat de seqüències de 30 nucleòtids utilitzades com a diana no són el lloc real d'unió del miRNA. Una altra problemàtica és la manca de repositoris que incloguin les dades de dianes que s'hagi comprovat experimentalment com a negatives. Molts softwares de predicció de dianes actuals utilitzen seqüències de miRNA generades artificialment (mock miRNA)[68, 69]. S'ha descartat aquesta opció per evitar que el model aprengui a discriminar aquests miRNA artificials dels reals, ignorant la resta de característiques.

Tot i els bons resultats obtinguts amb l'algorisme, s'han detectat punts de millora. En primer lloc, el treball s'ha limitat a dianes que segueixen una estructura llavor molt concreta, cosa que ha fet que determinades dianes que segueixen *bindings* no canònics hagin quedat fora del model. La dificultat en determinar possibles punts d'unió no canònics en la seqüència del 3'UTR ha marcat aquesta decisió. Les dades utilitzades en aquest TFM no contenen la seqüència exacte del mRNA diana que s'uneix al miRNA, de manera que s'ha hagut de buscar possibles punts d'unió, que poden ser o no, el veritable punt

d’anclatge. Tant pels resultats obtinguts, com pels resultats obtinguts per altres eines de predicció de dianes, els descriptors habituals d’aquestes eines semblen suficients per a avaluar dianes canòniques, però presenten limitacions per definir les dianes no canòniques.

A més, només s'ha valorat una possible diana en cada parella miRNA:3'UTR. La valoració de diferents punts d'unió entre la seqüència llavor i el 3'UTR podria millorar les prediccions. Només s'han tingut en compte 4 tipus de llavors canòniques, permetent aparellaments G:U, i no s'han valorat llavors no canòniques. Aquesta decisió s'ha pres tenint en compte les limitacions temporals i la complexitat que suposa buscar i analitzar aquest tipus de llavors. S'han pres algunes decisions relativament arbitràries, com la llargada de la seqüència diana a 30 nucleòtids, considerant que el els miRNA tenen una llargada d'uns 22 nucleòtids. No s'han explorat altres tècniques de reducció dimensional, a més de l'autoencoder, com el PCA o ICA, i finalment l'autoencoder no intervé en el model final.

Totes aquestes consideracions s'han de tenir en compte en futures versions de miRNAforest per tal de millorar la seva exactitud. Cal estendre el nombre de possibles dianes a avaluar en cada 3'UTR, ja que s'ha observat que la concentració de possibles dianes en un mRNA augmenta la probabilitat que sigui un objectiu del miRNA [70]. A més, s'ha de considerar ampliar el nombre de nucleòtids de les possibles dianes. Potser ampliar amb alguns nucleòtids en sentit 3' del mRNA pot millorar el model. S'ha de considerar l'ús d'altres tècniques de reducció dimensional i/o altres arquitectures d'autoencoder. Tenint en compte que la incorporació de la informació del tipus de llavor i el nombre d'aparellaments G:U ha millorat els models, cal explorar la incorporació d'altres descriptors, com la ubicació de la diana dins el 3'UTR [71] o l'abundància de dianes.

La metodologia plantejada inicialment és la que s'ha seguit, un desenvolupament iteratiu i creixent. Les dues fases del TFM han tingut una etapa d’iniciació, en que s’ha creat una primera versió de l’algorisme, simple i funcional, i una etapa d’iteració, en que l’algorisme s’ha redissenyat en diferents iteracions per obtenir-ne versions millorades. Aquesta metodologia s’ha demostrat com a encertada, ja que ha permès assolir els objectius. Evidentment, noves iteracions han de permetre millorar el rendiment del model. Pel que fa a la planificació temporal, l'obtenció d'un bon set de dades i la construcció de l'autoencoder han produït lleus desviacions en la previsió. Aquestes desviacions s'han resolt reduïnt el nombre d’iteracions en la construcció de l’autoencoder, centrant esforços en les arquitectures amb millors resultats en la reconstrucció de les dades originals.

Finalment, el disseny del software de predicció de dianes de miRNA ha permès aprofundir i ampliar coneixements tant en els llenguatges python i R com en diferents algorismes de Machine Learning. En la era de la informació, la biologia no queda al marge de la generació de grans quantitats de dades. En aquest sentit, es fa imprescindible ser capaç de processar i analitzar aquestes dades, cercant patrons que permetin fer prediccions. És per això que les tècniques de *Machine Learning* han guanyat una gran importància en la generació de models [72] i la classificació i diagnosi de malalties [73].

6. Glossari

3'UTR: Regió d'un mRNA madur que no codifica per amino-àcids i que es troba a l'extrem 3' de la molècula. Aquesta regió serveix de diana per als miRNA.

Algorisme: Conjunt d'instruccions o regles organitzades de forma lògica i ordenada que permeten solucionar un determinat problema

Capa: En xarxes neuronals artificials, cadascun dels conjunts de nodes (o unitats) que componen la xarxa. Les capes poden ser visibles, que contenen un conjunt de nodes que serveixen d'entrada a una capa oculta, on les unitats corresponen a les dades processades.

Corba ROC (Receiver Operating Characteristic): Representació gràfica de la raó de veritables positius (TPR: True Positive Rate) respecte la raó de falso positius (FPR: False Positive Rate) d'un model de classificació binari.

Dataset: Conjunt de dades que serveix per generar i avaluar un model amb tècniques de Machine Learning

Descriptor: Són cadascun dels atributs que descriuen una instància del conjunt de dades, equivalents a les variables estadístiques.

Ensemble learning: Tècnica de Machine Learning que utilitza múltiples algoritmes d'aprenentatge per obtenir millors prediccions que cadascun dels algorismes per separat.

Exemple: En machine learning, cadascun dels casos continguts en el dataset

Funció d'activació: En una ANN, cada node (o neurona) rep uns valors d'entrada, als quals se'ls aplica una funció, anomenada funció d'activació, per donar una senyal de sortida, que serviran com a capa visible de la següent capa de la xarxa.

Funció cost: Funció que calcula la distància (o desviació) de la solució obtinguda en una època respecte a la solució òptima en una ANN. A cada època de l'etapa d'aprenentatge, es calcula el cost del model obtingut, fins a obtenir la millor solució possible, és a dir, la que obté unes prediccions amb menys desviació dels valors reals.

Iteració: Acte de repetició un procés. El resultat de cada repetició, que també rep el nom d'iteració, serveix de punt de partida de la següent repetició

Machine learning: Conjunt d'algorismes que permeten identificar patrons complexes en gran quantitat de dades, amb l'objectiu de predir una característica o classificar noves dades.

miRNA: Cadenes curtes (aproximadament 22 nucleòtids) de RNA no codificant amb funció reguladora de l'expressió gènica a nivell post-transcripcional

mRNA: Cadena de RNA encarregada de portar fora del nucli la informació continguda en un gen de la cadena de DNA, per tal de sintetitzar la proteïna codificada en ell.

Node: Cadascuna de les unitats que formen part d'una capa en una xarxa neuronal artificial, el que equivaldria a una neurona en una xarxa neural natural.

Optimitzador: Aplicat a una xarxa neuronal, s'encarrega de buscar de forma iterativa en l'espai de paràmetres el conjunt que minimitza la funció cost, objectiu a l'hora d'entrenar un model.

Overfiting: L'objectiu en Machine Learning és generar un model general que tingui un bon comportament amb diferents sets de dades. L'overfitting és l'efecte d'ajustar en excés un model a les dades d'aprenentatge, de manera que és altament eficient en interpretar aquestes dades, però deficient a l'hora de fer prediccions en nous conjunts de dades.

Python: Llenguatge de programació altament utilitzat en el camp de la bioinformàtica.

Regularitzador: La regularització penalitza la complexitat d'un model, de manera que en ser incorporat a la funció de cost, permet reduir l'overfitting, evitant que un paràmetre tingui més pes que el conjunt de la resta

Seqüència llavor: Seqüència dels vuit primers nucleòtids de l'extrem 5' del miRNA, complementaris a la regió diana del transcript que regula.

Transcript: Cadascuna de les diferents cadenes de mRNA codificades en un gen.

7. Bibliografia

[1] Wolffe AP, Matzke MA. Epigenetics: regulation through repression. Science 1999; 286: 481-6.

[2] Bird, A. DNA methylation patterns and epigenetic memory. Genes Dev. 2002; 16: 6–21.

[3] Lennartsson A, Ekwall K. Histone modification patterns and epigenetic codes. Biochim Biophys Acta. 2009 Sep;1790(9):863-8.

[4] Becker PB, Workman JL. Nucleosome remodeling and epigenetics. Cold Spring Harb Perspect Biol. 2013 Sep 1;5(9).

[5] Daniel Holoch & Danesh Moazed. RNA-mediated epigenetic regulation of gene expression. Nature Reviews Genetics 2015; 16, 71–84.

[6] Jonas S. & Izaurralde E. Towards a molecular understanding of microRNA-mediated gene silencing. Nat Rev Genet 16, 421–433 (2015)

[7] Lewis BP., Burge CB., Bartel DP. Conserved Seed Pairing, Often Flanked by Adenosines, Indicates that Thousands of Human Genes are MicroRNA Targets. Cell, Vol. 120, 15–20 (2005)

[8] Jonas S, Izaurralde E. Towards a molecular understanding of microRNA-mediated gene silencing. Nature Reviews Genetics 16, 421–433 (2015)

[9] Relton CL, Davey Smith G. Epigenetic epidemiology of common complex disease: prospects for prediction, prevention, and treatment. PLoS Med. Oct 26;7(10):e1000356 (2010)

[10] Ranganna K, Mathew OP, Milton SG and Hayes BE. MicroRNAome of Vascular Smooth Muscle Cells: Potential for MicroRNA-Based Vascular Therapies. InTechOpen (2013)

[11] Peterson SM., Thompson JA., Ufkin ML., Sathyanarayana P., Liaw L., & Congdon CB. Common features of microRNA target prediction tools. Frontiers in Genetics, 5, 23 (2014)

[12] Kuzuoğlu-Öztürk D, Bhandari D, Huntzinger E, Fauser M, Helms S, Izaurralde E. miRISC and the CCR4–NOT complex silence mRNA targets independently of 43S ribosomal scanning. The EMBO Journal. Jun 1;35(11):1186-203 (2016)

[13] Lewis BP, Shih IH, Jones-Rhoades MW, Bartel DP, Burge CB. Prediction of mammalian microRNA targets. Cell 115, 787–798 (2003)

[14] Edward GA. Modern Thermodynamics by the Methods of Willard Gibbs. London: Methuen & Co. pg. 11 (1933)

[15] Lewis BP, Burge CB, Bartel DP. Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microRNA targets. Cell. Jan 14;120(1):15-20 (2005)

[16] Betel D, Koppal A, Agius P, Sander C, Leslie C. Comprehensive modeling of microRNA targets predicts functional non-conserved and non-canonical sites. Genome Biol. 11(8):R90 (2010)

[17] Kertesz M, Iovino N, Unnerstall U, Gaul U, Segal E. The role of site accessibility in microRNA target recognition. Nat. Genet., 39(10), 1278 (2007)

[18] Kiriakidou M, Nelson PT, Kouranov A, Fitziev P, Bouyioukos C, Mourelatos Z, Hatzigeorgiou A. A combined computational-experimental approach predicts human microRNA targets. Genes Dev. 18(10), 1165-1178 (2004)

[19] Rehmsmeier M,Steffen P, Hochsmann M, Giegerich R. Fast and effective prediction of microRNA/target duplexes. RNA, Oct;10(10):1507-17 (2004)

[20] Wang X and ElNaqa IM.Prediction of both conserved and non-conserved microRNA targets in animals. Bioinformatics 24, 325–332 (2008)

[21] Miranda KC,Huynh T, Tay Y, Ang YS, Tam WL, Thomson AM, Lim B & Rigoutsos I. A pattern-based method for the identification of MicroRNA binding sites and their corresponding heteroduplexes. Cell, 126(6):1203-17 (2006)

[22] Bandyopadhyay S and Mitra R. TargetMiner: microRNA target pre-diction with systematic identification of tissue-specific negative examples. Bioinformatics 25, 2625–2631 (2009)

[23] Coronnello C & Benos PV. ComiR: combinatorial microRNA target prediction tool. Nucleic Acids Res. 41, W159–W164 (2013)

[24] Lee B, Baek J, Park S, Yoon S. deepTarget: End-to-end Learning Framework for microRNA Target Prediction using Deep Recurrent Neural Networks. arXiv:1603.09123

[25] Rish I. An empirical study of the naive Bayes classifier. IJCAI Workshop on Empirical Methods in AI (2001)

[26] Jain AK, Mao J, Mohiuddin KM. Artificial neural networks: a tutorial. Computer (Volume: 29, Issue: 3, Mar 1996)

[27] Petersen JP. Artificial Neural Network. http://pypr.sourceforge.net/ann.html

[28] Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician. 46 (3): 175–185 (1992)

[29] Hilbe JM. Logistic Regression Models. Chapman & Hall/CRC Press (2009)

[30] Sayad S. Logistic Regression. http://www.saedsayad.com/logistic_regression.htm

[31] Utgoff PE. Incremental induction of decision trees. Machine learning, 4(2), 161-186 (1989)

[32] Breiman L. Random Forests. Machine Learning. 45 (1): 5–32 (2001)

[33] Nguyen C, Wang Y, Nguyen HN. Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic. Journal of Biomedical Science and Engineering 6, 551-560 (2013)

[34] Lee Y, Lin Y, Wahba G. Multicategory Support Vector Machines. Journal of the American Statistical Association. 99 (465): 67 (2004)

[35] Ruiz-Gonzalez R, Gomez-Gil J, Gomez-Gil FJ and Martínez-Martínez V. An SVM-Based Classifier for Estimating the State of Various Rotating Components in Agro-Industrial Machinery. Sensors 2014 Nov; 14(11): 20713–20735.

[36] Mason L, Baxter J, Bartlett P and Frean M. Boosting Algorithms as Gradient Descent, in S. A. Solla, T. K. Leen, and K.-R. Muller, editors, Advances in Neural Information Processing Systems 12, pp. 512-518, MIT Press (2000)

[37] Abdi H and Williams LJ. Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics. 2 (4): 433–459 (2010)

[38] Hyvärinen A and Oja E. Independent Component Analysis:Algorithms and Applications. Neural Networks. 4-5. 13 (4–5): 411–430 (2000)

[39] Hinton GE and Salakhutdinov RR. Reducing the Dimensionality of Data with Neural Networks. Science 28 Jul 2006: Vol. 313, Issue 5786, pp. 504-507

[40] Coates A, Lee H and Ng AY. An analysis of single-layer networks in unsupervised feature learning. International Conference on Artificial Intelligence and Statistics (AISTATS – 2011)

[41] DL4J. A Beginner's Tutorial for Restricted Boltzmann Machines . https://deeplearning4j.org/restrictedboltzmannmachine

[42] Hinton G. Deep belief networks. Scholarpedia. 4 (5): 5947 (2009)

[43] Coates A, Ng A, Lee H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, PMLR 15:215-223 (2011)

[44] Bengio Y. Deep Belief Networks. http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/DeepBeliefNetworks

[45] Ng A et al. Autoencoders. http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

[46] Berindan-Neagoe I., Monroig P., Pasculli B., & Calin GA. MicroRNAome genome: a treasure for cancer diagnosis and therapy. CA Cancer Journal for Clinicians, 64(5) (2014)

[47] Xiao F, Zuo Z, Cai G, Kang S, Gao X, Li T. miRecords: an integrated resource for microRNA-target interactions. Nucleic Acids Res. 2009 Jan;37(Database issue):D105-10

[48] Chou CH, Chang NW, Shrestha S, Hsu SD, Lin YL, Lee WH, Yang CD, Hong HC, Wei TY, Tu SJ, Tsai TR, Ho SY, Jian TY, Wu HY, Chen PR, Lin NC, Huang HT, Yang TL, Pai CY, Tai CS, Chen WL, Huang CY, Liu CC, Weng SL, Liao KW, Hsu WL, Huang HD. miRTarBase 2016: updates to the experimentally validated miRNA-target interactions database. Nucleic Acids Res. 2016 Jan 4;44(D1):D239-47

[49] Vlachos IS, Paraskevopoulou MD, Karagkouni D, Georgakilas G, Vergoulis T, Kanellos I, Anastasopoulos IL, Maniou S, Karathanou K, Kalfakakou D, Fevgas A, Dalamagas T, Hatzigeorgiou AG. DIANA-TarBase v7.0: indexing more than half a million experimentally supported miRNA:mRNA interactions. Nucleic Acids Res. 2015 Jan;43(Database issue):D153-9

[50] Kozomara A, Griffiths-Jones S. miRBase: annotating high confidence microRNAs using deep sequencing data. Nucleic Acids Res. 2014 Jan;42(Database issue):D68-73

[51] Lee YJ, Kim V, Muth D, Witwer KW. Validated MicroRNA Target Databases: an Evaluation. Drug development research. 76(7):389-396 (2015)

[52] John B, Enright AJ, Aravin A, Tuschl T, Sander C & Marks DS. Human microRNA targets. PLoS Biol, 2(11):e363 (2004)

[53] Menor M, Ching T, Zhu X, Garmire D, Garmire LX. mirMark: a site-level and UTR-level classifier for miRNA target prediction. Genome Biol. 15(10):500 (2014)

[54] Xiao F, Zuo Z, Cai G, Kang S, Gao X, Li T. miRecords: an integrated resource for microRNA-target interactions. Nucleic Acids Res. 37: D105-D110 (2009)

[55] Hsu SD, Tseng YT, Shrestha S, Lin YL, Khaleel A, Chou CH, Chu CF, Huang HY, Lin CM, Ho SY, Jian TY, Lin FM, Chang TH, Weng SL, Liao KW, Liao IE, Liu CC, Huang HD. miRTarBase update 2014: an information resource for experimentally validated miRNA-target interactions. Nucleic Acids Res. 42: D78-D85 (2014)

[56] Lewis BP, Shih IH, Jones-Rhoades MW, Bartel DP & Burge CB. Prediction of mammalian microRNA targets. Cell 115, 787–798 (2003)

[57] Lewis BP, Burge CB & Bartel DP. Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microRNA targets. Cell 120, 15–20 (2005)

[58] Brennecke J, Stark A, Russell RB & Cohen SM. Principles of microRNA-target recognition. PLoS Biol. 3:e85 (2005)

[59] Krek A, Grun D, Poy MN, Wolf R, Rosenberg L, Epstein EJ, MacMenamin P, da Piedade I, Gunsalus KC, Stoffel M, Rajewsky N. Combinatorial microRNA target predictions. Nat. Genet. 37, 495–500 (2005)

[60] Yue D, Liu H, Huang Y. Survey of Computational Algorithms for MicroRNA Target Prediction. Curr Genomics 10(7):478-92 (2009)

[61] Wikipedia. One-hot. https://en.wikipedia.org/wiki/One-hot

[62] Keras: Deep Learning library for Theano and TensorFlow. https://keras.io/

[63] Agostinelli F, Hoffman M, Sadowski P, Baldi P. Learning Activation Functions to Improve Deep Neural Networks. arXiv:1412.6830

[64] Lawrence J. Introduction to Neural Networks, California Scientific Software Press. (1994)

[65] Ruder S. An overview of gradient descent optimization algorithms. arXiv:1609.04747

[66] Alain G and Bengio Y. What Regularized Auto-Encoders Learn from the Data-Generating Distribution. Journal of Machine Learning Research 15, 3563-3593 (2014)

[67] Sancho Caparrini F. Entrenamiento de Redes Neuronales: mejorando el Gradiente Descendiente. http://www.cs.us.es/~fsancho/?e=165

[68] John B, Enright AJ, Aravin A, Tuschl T, Sander C, Marks DS. Human microRNA targets. PLoS Biol, 2(11):e363 (2004)

[69] Maragkakis M, Alexiou P, Papadopoulos G, Reczko M, Dalamagas T, Giannopoulos G, Goumas G, Koukis E, Kourtis K, Simossis VA. Accurate microRNA target prediction correlates with protein repression levels. BMC Bioinformatics, 10(1):295 (2009)

[70] Garcia DM, Baek D, Shin C, Bell GW, Grimson A, Bartel DP. Weak seed-pairing stability and high target-site abundance decrease the proficiency of lsy-6 and other microRNAs. Nat Struct Mol Biol. Sep 11;18(10):1139-46. (2011)

[71] Grimson A, Farh KK, Johnston WK, Garrett-Engele P, Lim LP, Bartel DP. MicroRNA targeting specificity in mammals: determinants beyond seed pairing. Mol Cell. Jul 6;27(1):91-105. (2007)

[72] Libbrecht MW, Noble WS. Machine learning applications in genetics and genomics. Nat Rev Genet. Jun;16(6):321-32. (2015)

[73] Khan J, Wei JS, Ringnér M, Saal LH, Ladanyi M, Westermann F, Berthold F, Schwab M, Antonescu CR, Peterson C, Meltzer PS. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nat Med. Jun;7(6):673-9. (2001)

8. Annexes

8.1. Scripts R

8.1.1. Generació del dataset

```
# Es fixa el directori de treball el directori on es troba l'arxiu .R
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

# PROCESSAT DE LES DADES POSITIVES
# Es llegeix l'arxiu .csv que conté les dades positives
positive <- read.csv("rawData/Positive_data.csv", header = TRUE)

require(biomaRt)
require(org.Hs.eg.db)

# S'obtenen les seq 3'UTR dels diferents transcripts

refseq <- unique(positive$mRNA_ID)
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
utr3_pos <- getSequence(id=refseq, type="refseq_mrna", seqType="3utr",
                        mart=ensembl)

# S'obté el nom del gen de cada transcript

gene_symbol <- getBM(attributes=c('hgnc_symbol', 'refseq_mrna'),
                filters = 'refseq_mrna',
                    values = refseq, mart = ensembl)
positive <- merge(positive, gene_symbol, by.x = "mRNA_ID",
                            by.y = "refseq_mrna")

# Es llegeix la seqüència dels miRNAs

library("seqinr")
mirna <- read.fasta("rawData/mature.fa")
hsa_mirna <- mirna[grepl("^hsa", names(mirna))]

miR_ID <- c()
miRNA_seq <- c()
for(i in 1:length(hsa_mirna)){
      miR_ID[i] <- attr(hsa_mirna[[i]], "name")
      seq <- c()
      for(j in 1:length(hsa_mirna[[i]])){
            seq[j] <- hsa_mirna[[i]][j]
```

```
        }
        miRNA_seq[i] <- paste(seq, sep="", collapse ="")
}
mirna_seq <- data.frame(cbind(miR_ID, miRNA_seq))

# Es creuen les dades, de manera que s'obté una taula amb cada
# combinació de miRNA i seq 3'UTR

positive.data <- merge(positive, utr3_pos, by.x = "mRNA_ID",
                                by.y = "refseq_mrna")
positive.data <- merge(positive.data, mirna_seq, by = "miR_ID")
positive.data <- positive.data[,c(1, 5, 3, 2, 4)]
colnames(positive.data)[3] <- "gene_symbol"

positive.data <- positive.data[!(positive.data$`3utr` == "Sequence
                    unavailable"),]
target <- rep(1, dim(positive.data)[1])
positive.data <- cbind(target, positive.data)

# Nombre de transcripts
length(unique(positive.data$mRNA_ID))

# Nombre de gens
length(unique(positive.data$gene_symbol))

# Nombre de miRNA
length(unique(positive.data$miR_ID))

# PROCESSAT DE LES DADES NEGATIVES

# Es llegeix l'arxiu .csv que conté les dades negatives

negative <- read.csv("rawData/negativeTranscriptsTarbase.csv",
                                sep = ";", header = TRUE)

require(biomaRt)
require(org.Hs.eg.db)

# S'obtenen les seq 3'UTR dels diferents transcripts

ensemblID <- unique(negative$EnsemblId)
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
utr3_neg <- getSequence(id=ensemblID, type="ensembl_gene_id",
              seqType="3utr", mart=ensembl)
```

```
negative <- merge(negative, utr3_neg, by.x = "EnsemblId",
                          by.y = "ensembl_gene_id")

# Es llegeix la seqüència dels miRNAs

library("seqinr")
mirna <- read.fasta("rawData/mature.fa")
hsa_mirna <- mirna[grepl("^hsa", names(mirna))]

miR_ID <- c()
miRNA_seq <- c()
for(i in 1:length(hsa_mirna)){
      miR_ID[i] <- attr(hsa_mirna[[i]], "name")
      seq <- c()
      for(j in 1:length(hsa_mirna[[i]])){
            seq[j] <- hsa_mirna[[i]][j]
      }
      miRNA_seq[i] <- paste(seq, sep="", collapse ="")
}
mirna_seq <- data.frame(cbind(miR_ID, miRNA_seq))

# Es creuen les dades, de manera que s'obté una taula amb cada
# combinació de miRNA i seq 3'UTR

negative.data <- merge(negative, mirna_seq, by.x = "miRNA",
                                  by.y = "miR_ID")
negative.data <- negative.data[,c(4, 1, 6, 3, 2, 5)]

# S'eliminen exemples que no tenen seq de 3'UTR o seq massa curtes

negative.data <- negative.data[!(negative.data$`3utr` == "Sequence
                     unavailable"),]
negative.data <- negative.data[!(nchar(negative.data$`3utr`) < 22),]

# Nombre de transcripts
length(unique(negative.data$`3utr`))

# Nombre de gens
length(unique(negative.data$gene_name))

# Nombre de miRNA
length(unique(negative.data$miRNA))

# Fusió de les dades positives i negatives
```

```
colnames(negative.data) <- colnames(positive.data)
dataset <- rbind(positive.data, negative.data)
dataset$target <- as.factor(dataset$target)
dataset$miR_ID <- as.character(dataset$miR_ID)
dataset$miRNA_seq <- as.character(dataset$miRNA_seq)
dataset$miRNA_seq <- toupper(dataset$miRNA_seq)

save(dataset, negative.data, positive.data, file = "1_dataset.RData")
rm(list=ls())
load(file = "1_dataset.RData")
```

8.1.2. Obtenció seqüència diana

```
# Es fixa el directori de treball el directori on es troba l'arxiu .R
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

load(file = "1_dataset.RData")
library(Biostrings)

# Es generen diferents vectors on es guradaran les dades generades

totalRegion <- rep(NA, dim(dataset)[1])
match_8mer <- rep(0, dim(dataset)[1])
match_7mer_m8 <- rep(0, dim(dataset)[1])
match_7mer_A1 <- rep(0, dim(dataset)[1])
match_6mer <- rep(0, dim(dataset)[1])
GU_wobble <- rep(0, dim(dataset)[1])

# Per mitjà d'un 'for loop' es buscarà una seqüència diana per a cada # miRNA

for(i in 1:dim(dataset)[1]){

    # Es canvien les T per U de la seqüència dels 3'UTR

  target <- dataset$`3utr`[i]
  target <- gsub('T', 'U', target)
  miRNAseq <- dataset$miRNA_seq[i]

    # Es generen les seqüències dels diferents tipus de llavor a        # considerar

  mer6 <- RNAString(substr(miRNAseq, 2, 7))
  mer6_rev <- as.character(reverseComplement(mer6))
  mer7_A1_rev <- paste0("A", mer6_rev)
  mer7_m8 <- RNAString(substr(miRNAseq, 2, 8))
  mer7_m8_rev <- as.character(reverseComplement(mer7_m8)
  mer8_rev <- paste0("A", mer7_m8_rev)

    # Es busca el primer tipus de llavor, el que té major                #
complementarietat. Si troba una seqüència complementària,          # selecciona
aquesta seqüència més els 22 nucleòtids e sentit        # 5'. Si la seqüència no té 30
nucleòtids, es descarta.

  if(grepl(mer8_rev, target)){
    pos <- gregexpr(mer8_rev, target)[[1]]
    pos <- pos[pos > 12]
```

```
        pos <- pos[pos+17 < nchar(target)]
        if(length(pos)>0){
            match_8mer[i] <- 1
            pos <- pos[1]
            totalRegion[i] <- substr(target, pos-12,
                                     pos+17)
        }
      # Si s'ha trobat una potencial diana, es busca un altre tipus de llavor

}else{
            # De la mateixa manera que en el cas anterior, es busca la llavor,
            # i si es troba una seqüència complementaria i es pot generar una
            # seqüència de 30 nucleòtid, es guarda, si no, es busquen els següents
            # tipus de llavor

        if(grepl(mer7_m8_rev, target)){
            pos <- gregexpr(mer7_m8_rev, target)[[1]]
            pos <- pos[pos > 13]
            pos <- pos[pos+16 < nchar(target)]
            if(length(pos)>0){
                match_7mer_m8[i] <- 1
                pos <- pos[1]
                totalRegion[i] <- substr(target, pos-13, pos+16)
            }
        }else{
            if(grepl(mer7_A1_rev, target)){
                pos <- gregexpr(mer7_A1_rev, target)[[1]]
                pos <- pos[pos > 13]
                pos <- pos[pos+16 < nchar(target)]
                if(length(pos)>0){
                    match_7mer_A1[i] <- 1
                    pos <- pos[1]
                    totalRegion[i]<-substr(target, pos-13, pos+16)
                }
            }else{
                if(grepl(mer6_rev, target)){
                    pos <- gregexpr(mer6_rev,
                                    target)[[1]]
                    pos <- pos[pos > 14]
                    pos<-pos[pos+15<nchar(target)]
                    if(length(pos)>0){
                        match_6mer[i] <- 1
                        pos <- pos[1]
                    totalRegion[i]<-substr(target, pos-14, pos+15)
```

```
                    }
                }else{
                    totalRegion[i] <- NA
                }
            }
        }
    }
        # Un cop s'han avaluat els 4 tipus de llavor sense èxit, es
        # permetran aparellaments G:U

    if(is.na(totalRegion[i])){
        mer6_rev_GU <- gsub('A', '[AG]', mer6_rev)
        mer6_rev_GU <- gsub('C', '[CU]', mer6_rev_GU)
        mer7_A1_rev_GU <- paste0(mer6_rev_GU, "A")
        mer7_m8_rev_GU <- gsub('A', '[AG]', mer7_m8_rev)
        mer7_m8_rev_GU <- gsub('C', '[CU]', mer7_m8_rev_GU)
        mer8_rev_GU <- paste0(mer7_m8_rev_GU, "A")

# Es busquen els diferents tipus de llavor en el mateix ordre que
# anteriorment. Cada llavor es guarda i es guarda el nombre
# d'aparellaments G:U

        GU_dist <- c()
        coordinate <- c()
        theoricalRegion <- c()
        if(grepl(mer8_rev_GU, target)){
            pos <- gregexpr(mer8_rev_GU, target)[[1]]
            pos <- pos[pos > 12]
            pos <- pos[pos+17 < nchar(target)]
            if(length(pos)>0){
                dist <- c()
                for(j in 1:length(pos)){
                    theorical_target<-substr(target,
                            pos[j], pos[j]+7)
                    dist[j] <- adist(mer8_rev,
                            theorical_target)[1]
                }
                GU_dist[1] <- dist[which.min(dist)]
                coordinate[1] <- pos[which.min(dist)]
                theoricalRegion[1] <- substr(target,
         coordinate[1]-12, coordinate[1]+17)
            }
        }
        if(grepl(mer7_m8_rev_GU, target)){
```

```
        pos <- gregexpr(mer7_m8_rev_GU, target)[[1]]
        pos <- pos[pos > 13]
        pos <- pos[pos+16 < nchar(target)]
        if(length(pos)>0){
              dist <- c()
              for(j in 1:length(pos)){
                   theorical_target<-substr(target,
                              pos[j], pos[j]+6)
                   dist[j] <- adist(mer7_m8_rev,
                              theorical_target)[1]
              }
             GU_dist[2] <- dist[which.min(dist)]
             coordinate[2] <- pos[which.min(dist)]
             theoricalRegion[2] <- substr(target,
 coordinate[2]-13, coordinate[2]+16)
        }
}
if(grepl(mer7_A1_rev_GU, target)){
        pos <- gregexpr(mer7_A1_rev_GU, target)[[1]]
        pos <- pos[pos > 13]
        pos <- pos[pos+16 < nchar(target)]
        if(length(pos)>0){
              dist <- c()
              for(j in 1:length(pos)){
                   theorical_target<-substr(target,
                              pos[j], pos[j]+6)
                   dist[j] <- adist(mer7_A1_rev,
                              theorical_target)[1]
              }
             GU_dist[3] <- dist[which.min(dist)]
             coordinate[3] <- pos[which.min(dist)]
             theoricalRegion[3] <- substr(target,
 coordinate[3]-13, coordinate[3]+16)
        }
}
if(grepl(mer6_rev_GU, target)){
        pos <- gregexpr(mer6_rev_GU, target)[[1]]
        pos <- pos[pos > 14]
        pos <- pos[pos+15 < nchar(target)]
        if(length(pos)>0){
              dist <- c()
              for(j in 1:length(pos)){
                   theorical_target<-substr(target,
                              pos[j], pos[j]+5)
```

```
                        dist[j] <- adist(mer6_rev,
                                theorical_target)[1]
                    }
                GU_dist[4] <- dist[which.min(dist)]
                coordinate[4] <- pos[which.min(dist)]
                theoricalRegion[4] <- substr(target,
            coordinate[4]-14, coordinate[4]+15)
                }
            }
            if(!is.null(coordinate)){
                # Finalment, es conserva la seqüència amb menor
                # nombre d'aparellaments G:U
            totalRegion[i]<-theoricalRegion[which.min(GU_dist)]
            GU_wobble[i]<-as.numeric(GU_dist[which.min(GU_dist)])

                if(which.min(GU_dist) == 1){match_8mer[i]<-1}
                if(which.min(GU_dist) == 2){match_7mer_m8[i]<-1}
                if(which.min(GU_dist) == 3){match_7mer_A1[i]<-1}
                if(which.min(GU_dist) == 4){match_6mer[i]<-1}
            }
        }
}

# Es normalitza el nombre d'aparellaments G:U, de manera que totes les
# variables es trobin entre 0 i 1

GU_wobble <- sapply(GU_wobble, function(x) (x - min(GU_wobble)) /
                                (max(GU_wobble) – min(GU_wobble)))

# S'afegeix la informació generada al dataset

totalRegionData <- as.data.frame(cbind(match_8mer, match_7mer_m8,
                        match_7mer_A1, match_6mer,
GU_wobble, totalRegion))
data <- dataset[, -6]
data <- cbind(data, totalRegionData[, c(6,1:5)])
data$totalRegion <- as.character(data$totalRegion)

# S'eliminen els exemples que contenen NA, corresponents als exemples
# en que no s'ha trobat cap seqüència diana en el 3'UTR

sum(is.na(data[data$target == 0,]))
dim(data[data$target == 0,])
sum(is.na(data[data$target == 1,]))
```

```
dim(data[data$target == 1,])

data <- na.omit(data)
dim(data[data$target == 0,])
dim(data[data$target == 1,])

save(data, file = "2_target.RData")
rm(list=ls())
load(file = "2_target.Rdata")
```

8.1.3. Codificació One-Hot

```
# Es fixa el directori de treball el directori on es troba l'arxiu .R
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

load(file = "2_target.RData")

# Es defineix la funció oneHot, que implementarà la codificació
# one-hot encoding

oneHot <- function(x, length){

    # En primer lloc, la funció incorpora una llista que ens
    # servirà de diccionari. En aquest llista, cada nucleòtid
    # portarà associat un one-hot encoding. El caràcter 'N'
    # servirà per omplir els nucleòtids buits

    ntDict <- vector(mode="list", length=5)
    names(ntDict) <- c("A", "U", "C", "G", "N")

    # Es generen els 5 one-hot encoding

    ntbin <- rep(0,4)
    ntDict[[5]] <- ntbin
    for(i in 1:4){
        nt <- ntbin
        nt[i] <- 1
        ntDict[[i]] <- nt
    }

    # La seqüència d'entrada es divideix en els diferents
    # caràcters, i s'afegeix el caràcter 'N' que ens servirà
    # per omplir els nucleòtids absents

    x <- toupper(x)
    ntseq <- strsplit(x, split="")[[1]]

    if(length(ntseq) < length){
        ntseq[(length(ntseq)+1):length] <- "N"
    }

    # La funció crea una nova llista, on es guarda el one-hot
    # encoding del miRNA
```

```
        encodelist <- vector(mode="list", length=length(ntseq))
        for(i in 1:length(ntseq)){
                encodelist[[i]]<- ntDict[[ntseq[i]]]
        }

        # Finalment, la seqüència queda codificada amb one-Hot
        # encoding

        as.vector(unlist(encodelist))
}

# En primer lloc, codifiquem els miRNA del dataset

mirna_seq <- data$miRNA_seq
SeqsEncode <- vector(mode="list", length=length(mirna_seq))
names(SeqsEncode) <- mirna_seq

# Tots els codis resultants tindran la mida del miRNA més llarg

len <- max(nchar(mirna_seq))
for(i in 1:length(mirna_seq)){
        SeqsEncode[[i]] <- oneHot(mirna_seq[i], len)
}

miRNAseqMatrix <- t(as.data.frame(SeqsEncode))
miRNA_oneHot <- as.data.frame(miRNAseqMatrix)
names(miRNA_oneHot) <- paste0("miR_OH", 1:length(names(miRNA_oneHot)))
rownames(miRNA_oneHot) <- NULL

# codifiquem els mRNA del dataset

totalRegion_seq <- data$totalRegion
SeqsEncode <- vector(mode="list", length=length(totalRegion_seq))
names(SeqsEncode) <- totalRegion_seq

len <- max(nchar(totalRegion_seq))
for(i in 1:length(totalRegion_seq)){
        SeqsEncode[[i]] <- oneHot(totalRegion_seq[i], len)
}

mRNAseqMatrix <- t(as.data.frame(SeqsEncode))
mRNA_oneHot <- as.data.frame(mRNAseqMatrix)
rownames(mRNA_oneHot) <- NULL
names(mRNA_oneHot) <- paste0("mRNA_OH", 1:length(names(mRNA_oneHot)))
```

```
data <- cbind(data, miRNA_oneHot, mRNA_oneHot)

save(data, file = "3_oneHot_encode.RData")
rm(list=ls())
load(file = "3_oneHot_encode.RData")
```

8.1.4. Divisió de les dades

```
# Es fixa el directori de treball el directori on es troba l'arxiu .R
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

load(file = "3_oneHot_encode.RData")

# Es fixa una llavor que asseguri la reproductibilitat
set.seed(1980)

# Es divideix el dataset en dos datasets, de manera que tinguin el
# mateix nombre de dades positives i negatives entre si

negData <- data[data$target == 0,]
posData <- data[data$target == 1,]
negIndex <- sample(1:dim(negData)[1], dim(negData)[1]/2)
posIndex <- sample(1:dim(posData)[1], dim(posData)[1]/2)

# Es genera el dataset que servirà per descobrir nous descriptors,
# dividit en el subset 'train' i el subset 'test'

negData_features <- negData[negIndex,]
posData_features <- posData[posIndex,]
featuresData <- rbind(negData_features, posData_features)
trainNegIndex <- sample(1:dim(negData_features)[1],
                        round(dim(negData_features)[1]*.75))
trainPosIndex <- sample(1:dim(posData_features)[1],
                        round(dim(posData_features)[1]*.75))
trainFeatures <- rbind(negData_features[trainNegIndex,],
          posData_features[trainPosIndex,])
testFeatures <- rbind(negData_features[-trainNegIndex,],
          posData_features[-trainPosIndex,])

write.csv(trainFeatures[,12:231], file = "data/trainFeatures.csv",
          quote = FALSE, row.names = FALSE)
write.csv(testFeatures[,12:231], file = "data/testFeatures.csv",
          quote = FALSE, row.names = FALSE)

# Es genera el dataset que servirà per predir dianes de miRNA,
# dividit en el subset 'train' i el subset 'test'

negData_predict <- negData[-negIndex,]
posData_predict <- posData[-posIndex,]
predictData <- rbind(negData_predict, posData_predict)
```

```
trainNegIndex <- sample(1:dim(negData_predict)[1],
                        round(dim(negData_predict)[1]*.75))
trainPosIndex <- sample(1:dim(posData_predict)[1],
                        round(dim(posData_predict)[1]*.75))

trainPredict <- rbind(negData_predict[trainNegIndex,],
          posData_predict[trainPosIndex,])
testPredict <- rbind(negData_predict[-trainNegIndex,],
          posData_predict[-trainPosIndex,])

write.csv(trainPredict, file = "data/trainPredict.csv",
          quote = FALSE, row.names = FALSE)
write.csv(testPredict, file = "data/testPredict.csv",
          quote = FALSE, row.names = FALSE)

save(data, featuresData, predictData, file = "4_subsettingData.RData")
rm(list=ls())
load(file = "4_subsettingData.RData")
```

8.2. Scripts Python

8.2.1. Autoencoder: exemple de 60 nodes ocults

```
import pandas as pd
import numpy as np
np.random.seed(1980) # S'ajusta llavor per tal que el resultat sigui reproduible
from keras.layers import Input, Dense
from keras.models import Model
from keras import regularizers
from keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt

# S'estableix que l'aprenentatge s'aturi si en 10 epoques el model no # millora

early_stopping = EarlyStopping(monitor='val_loss', patience=10)

# Es carreguen les dades per entrenar i testar el model

train = pd.read_csv('Data/trainFeatures.csv')
train = np.array(train)
test = pd.read_csv('Data/testFeatures.csv')
test = np.array(test)

# Es construeix l'autoencoder. S’avaluen les diferents funcions
# d'activacio: 'relu', 'softmax', 'elu', 'softplus', 'softsign',
# 'tanh', 'sigmoid', 'hard_sigmoid', 'linear'

inputs = Input(shape=(220,))
encoded = Dense(60, activation='softplus',
kernel_regularizer=regularizers.l2(1e-5))(inputs)
outputs = Dense(220, activation='sigmoid')(encoded)

# Es defineix el model, aixi com la funcio 'loss' que permetra
# l'aprenentatge

encoder = Model(inputs, encoded)
autoencoder = Model(input=inputs, output=outputs)
encoder.compile(optimizer='Nadam', loss='mean_squared_error')
autoencoder.compile(optimizer='Nadam', loss='mean_squared_error')

# S'entrena el model i es guarda per ser utilitzat més endavant

autoencoder.fit(train, train,
            epochs=1000,
            shuffle=True,
            validation_data=(test, test),
            callbacks=[early_stopping])
encoder.save('Models/encoder_H60.h5', overwrite=True)
```

```
# Es guarda el model en format YAML

encoder_H60 = encoder.to_yaml()
with open("Models/encoder_H60.yaml", "w") as yaml_file:
    yaml_file.write(encoder_H60)

# Es fan les prediccions i es guarden en un arxiu .csv per ser
# evaluades posteriorment

predictions = autoencoder.predict(test)
np.savetxt("Results/Hidden60/decoded_softplus.csv", predictions, delimiter=",")

# Es redueixen dimensionalment les dades

encoded_data = encoder.predict(test)
print encoded_data.shape

# Es representa graficament l'evolucio de l'error durant l'entrenament
plt.figure(figsize=(6, 3))
plt.plot(autoencoder.history.history['loss'])
plt.ylabel('error')
plt.xlabel('iteration')
plt.title('training error')
plt.show()
```

8.2.2. Algoritme de predicció: Random Forest

```
import pandas as pd
import numpy as np
np.random.seed(1980) # S'ajusta llavor per tal que el resultat sigui reproduible
from keras.models import load_model
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score
from sklearn.metrics import f1_score

# Es carreguen les dades train i es passen per l'encoder. El model
# final no utilitza l'autoencoder, es manté per a futures iteracions

encoder = load_model('autoencoder/encoder_H20.h5')

# Es carreguen les dades train i es passen per l'encoder. El model
# final no utilitza l'autoencoder, es manté per a futures iteracions

train = pd.read_csv('data/trainPredict.csv')
train = np.array(train)
train_onehot = train[:, 11:]
## encoded_train = encoder.predict(train_onehot)

# S'afegeix la informacio referent a la llavor

encoded_train = np.column_stack((train[:, 6:11], train_onehot))

# Es carreguen les etiquetes del subset train

y_train = train[:, 0]
y_train = y_train.astype("float64")

# Es carreguen les dades test i es passen per l'encoder. El model
# final no utilitza l'autoencoder, es manté per a futures iteracions

test = pd.read_csv('data/testPredict.csv')
test = np.array(test)
test_onehot = test[:, 11:]
## encoded_test = encoder.predict(test_onehot)

# S'afegeix la informacio referent a la llavor

encoded_test = np.column_stack((test[:, 6:11], test_onehot))

# Es crea el model amb l'algorisme Random Forest

model= RandomForestClassifier(n_estimators=35, bootstrap=False)

# S'entrena el model amb el subset train
```

```
model.fit(encoded_train, y_train)

# Es guarda el model per ser utilitzar posteriorment

import pickle
filename = 'models/RF_model.sav'
pickle.dump(model, open(filename, 'wb'))
# Estimem la precisio del model per cross-validation

scores = cross_val_score(model, encoded_train, y_train, cv=10)
print scores
print("Accuracy estimation: %0.4f (+/- %0.4f)" % (scores.mean(),
        scores.std() * 2))

# Es prediuen les categories dels exemples del subset test i
# es guarden per ser avaluades posteriorment

predicted = model.predict(encoded_test)
np.savetxt("results/RF_seed.csv", predicted, delimiter=",")

# Es calcula l'exactitud i el F1-score del model

y_test = test[:, 0]
y_test = y_test.astype("float64")
result = model.score(encoded_test, y_test)
print("Accuracy: %0.4f" % (result))

F1 = f1_score(y_test, predicted, pos_label=1)
print F1
```

8.2.3. miRNAforest.py: software de predicció

```
from Rfunctions import targetSeq, oneHot
import numpy as np

# Introduir la sequencia del miRNA i del 3UTR que es vulgui testar

miRNA = 'sequencia del miRNA'
UTR = 'sequencia del 3UTR'

# El programa busca possibles punts d'unio al 3UTR, gracies a la
# funció continguda en el fitxer Rfunctions.R contingut en el mateix
# directori. Si no en troba cap, s'atura, si en troba alguna, prediu
# si és una possible diana

mRNASeq = targetSeq(miRNA, UTR)

if(np.isnan(mRNASeq[5])):
   print 'Canonical seed not found'

else:

   # Es generen els codis one-Hot de les dues sequencies

   oneHotSeq = oneHot(miRNA, mRNASeq)
   target_oneHot = np.array(oneHotSeq)

   # S'importa el model generat anteriorment i s'avalua la possible
   # diana

   import pickle
   filename = 'RF_model.sav'
   RF = pickle.load(open(filename, 'rb'))
   result = RF.predict(target_oneHot)

   # El programa dona un veredicte possitiu o negatiu

   if result == 0.:
      print "Your 3'UTR is NOT a Target of your miRNA"

   elif result == 1.:
      print "Your 3'UTR is a Target of your miRNA"
```

8.2.3. Rfunctions.R: funcions per executar codi R amb Python

```
import rpy2.robjects as robjects

# Es generen els objectes R que permetran que python executi codi R.
# Es creen dues funcions, una per a buscar la possible sequencia
# d'unió del 3UTR i la segona que conte el 'one-Hot encoding'

robjects.r('''
  targetSeq <- function(miRNAseq, utr){

      library(Biostrings)

      totalRegion <- NA
      match_8mer <- 0
      match_7mer_m8 <- 0
      match_7mer_A1 <- 0
      match_6mer <- 0
      GU_wobble <- 0

      target <- gsub('T', 'U', utr)

      mer6 <- RNAString(substr(miRNAseq, 2, 7))
      mer6_rev <- as.character(reverseComplement(mer6))
      mer7_A1_rev <- paste0("A", mer6_rev)
      mer7_m8 <- RNAString(substr(miRNAseq, 2, 8))
      mer7_m8_rev <- as.character(reverseComplement(mer7_m8))
      mer8_rev <- paste0("A", mer7_m8_rev)

      if(grepl(mer8_rev, target)){
            pos <- gregexpr(mer8_rev, target)[[1]]
            pos <- pos[pos > 12]
            pos <- pos[pos+17 < nchar(target)]
            if(length(pos)>0){
                  match_8mer <- 1
                  pos <- pos[1]
                  totalRegion <- substr(target, pos-12, pos+17)
            }
      }else{
            if(grepl(mer7_m8_rev, target)){
                  pos <- gregexpr(mer7_m8_rev, target)[[1]]
                  pos <- pos[pos > 13]
                  pos <- pos[pos+16 < nchar(target)]
                  if(length(pos)>0){
                        match_7mer_m8 <- 1
                        pos <- pos[1]
                        totalRegion <- substr(target, pos-13,
                                              pos+16)
                  }
```

```
        }else{
            if(grepl(mer7_A1_rev, target)){
                pos<-gregexpr(mer7_A1_rev, target)[[1]]
                pos <- pos[pos > 13]
                pos <- pos[pos+16 < nchar(target)]
                if(length(pos)>0){
                    match_7mer_A1 <- 1
                    pos <- pos[1]
                    totalRegion<-substr(target,
                                pos-13, pos+16)
                 }
            }else{
                if(grepl(mer6_rev, target)){
                   pos<-gregexpr(mer6_rev, target)[[1]]
                   pos <- pos[pos > 14]
                   pos <- pos[pos+15 < nchar(target)]
                      if(length(pos)>0){
                         match_6mer <- 1
                         pos <- pos[1]
                         totalRegion<-substr(target,
                                          pos-14, pos+15)
                       }
                  }else{
                        totalRegion <- NA
                  }
            }
        }
}
if(is.na(totalRegion)){
        mer6_rev_GU <- gsub('A', '[AG]', mer6_rev)
        mer6_rev_GU <- gsub('C', '[CU]', mer6_rev_GU)
        mer7_A1_rev_GU <- paste0(mer6_rev_GU, "A")
        mer7_m8_rev_GU <- gsub('A', '[AG]', mer7_m8_rev)
        mer7_m8_rev_GU <- gsub('C', '[CU]', mer7_m8_rev_GU)
        mer8_rev_GU <- paste0(mer7_m8_rev_GU, "A")

        GU_dist <- c()
        coordinate <- c()
        theoricalRegion <- c()
        if(grepl(mer8_rev_GU, target)){
            pos <- gregexpr(mer8_rev_GU, target)[[1]]
            pos <- pos[pos > 12]
            pos <- pos[pos+17 < nchar(target)]
            if(length(pos)>0){
               dist <- c()
               for(j in 1:length(pos)){
                    theorical_target<-substr(target,
                                pos[j], pos[j]+7)
                    dist[j]<-adist(mer8_rev,
                      theorical_target)[1]
```

```
            }
        GU_dist[1] <- dist[which.min(dist)]
        coordinate[1] <- pos[which.min(dist)]
        theoricalRegion[1]<-substr(target,
                                   coordinate[1]-12,
                           coordinate[1]+17)
      }
}
if(grepl(mer7_m8_rev_GU, target)){
      pos <- gregexpr(mer7_m8_rev_GU, target)[[1]]
      pos <- pos[pos > 13]
      pos <- pos[pos+16 < nchar(target)]
      if(length(pos)>0){
         dist <- c()
         for(j in 1:length(pos)){
             theorical_target<-substr(target,
                    pos[j], pos[j]+6)
             dist[j] <- adist(mer7_m8_rev,
                           theorical_target)[1]
            }
          GU_dist[2] <- dist[which.min(dist)]
          coordinate[2] <- pos[which.min(dist)]
          theoricalRegion[2]<-substr(target,
                                     coordinate[2]-13,
                                     coordinate[2]+16)
      }
}
if(grepl(mer7_A1_rev_GU, target)){
    pos <- gregexpr(mer7_A1_rev_GU, target)[[1]]
    pos <- pos[pos > 13]
    pos <- pos[pos+16 < nchar(target)]
    if(length(pos)>0){
        dist <- c()
        for(j in 1:length(pos)){
            theorical_target<-substr(target,
                    pos[j], pos[j]+6)
            dist[j] <- adist(mer7_A1_rev,
                     theorical_target)[1]
             }
        GU_dist[3] <- dist[which.min(dist)]
        coordinate[3] <- pos[which.min(dist)]
        theoricalRegion[3]<-substr(target,
                           coordinate[3]-13,
                           coordinate[3]+16)
      }
}
if(grepl(mer6_rev_GU, target)){
    pos <- gregexpr(mer6_rev_GU, target)[[1]]
    pos <- pos[pos > 14]
    pos <- pos[pos+15 < nchar(target)]
```

```
            if(length(pos)>0){
                dist <- c()
                for(j in 1:length(pos)){
                    theorical_target <- substr(target,
                            pos[j], pos[j]+5)
                    dist[j] <- adist(mer6_rev,
                                     theorical_target)[1]
                     }
                GU_dist[4] <- dist[which.min(dist)]
                coordinate[4] <- pos[which.min(dist)]
                theoricalRegion[4] <- substr(target,
                                     coordinate[4]-14,
                                     coordinate[4]+15)
              }
          }
          if(!is.null(coordinate)){
              totalRegion<-theoricalRegion[which.min(GU_dist)]
              GU_wobble <- GU_dist[which.min(GU_dist)]

              if(which.min(GU_dist) == 1){match_8mer <- 1}
              if(which.min(GU_dist) == 2){match_7mer_m8 <- 1}
              if(which.min(GU_dist) == 3){match_7mer_A1 <- 1}
              if(which.min(GU_dist) == 4){match_6mer <- 1}
                }
      }

      GU_wobble <- sapply(GU_wobble, function(x) x/5)
      totalRegionData <- as.data.frame(cbind(match_8mer,
                                     match_7mer_m8, match_7mer_A1,
                                                match_6mer, GU_wobble,
                                                totalRegion))
      return(totalRegionData)
}

oneHot <- function(miRNA, target){

      ntDict <- vector(mode="list", length=5)
      names(ntDict) <- c("A", "U", "C", "G", "N")

      ntbin <- rep(0,4)
      ntDict[[5]] <- ntbin
      for(i in 1:4){
            nt <- ntbin
            nt[i] <- 1
            ntDict[[i]] <- nt
      }

      # miRNA
      miRNA <- as.character(miRNA)
      ntseq <- strsplit(miRNA, split="")[[1]]
```

```
        if(length(ntseq) < 25){
            ntseq[(length(ntseq)+1):25] <- "N"
        }

        encodelist <- vector(mode="list", length=length(ntseq))
        for(i in 1:length(ntseq)){
            encodelist[[i]]<- ntDict[[ntseq[i]]]
        }

        mirnaOneHot <- as.vector(unlist(encodelist))

        # target
        totalRegion <- as.character(target[[6]])
        ntseq <- strsplit(totalRegion, split="")[[1]]

        encodelist <- vector(mode="list", length=length(ntseq))
        for(i in 1:length(ntseq)){
            encodelist[[i]]<- ntDict[[ntseq[i]]]
        }

        targetOneHot <- as.vector(unlist(encodelist))
        seed <- as.numeric(as.vector(unlist(target[1:5])))

        t(c(seed, mirnaOneHot, targetOneHot))
}
        ''')

targetSeq = robjects.r['targetSeq']
oneHot = robjects.r['oneHot']
```