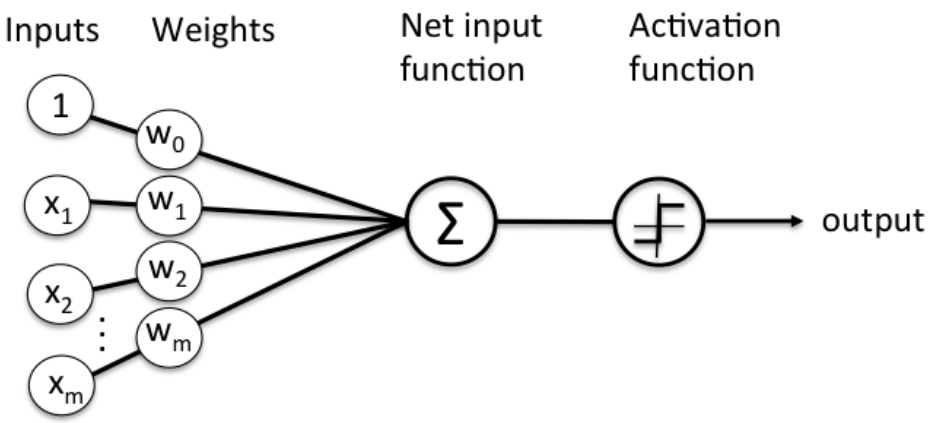
2n Parcial APC

Bloc 4: Xarxes Neurals

1. Descriu el perceptró de Frank Rosenblatt, i com una Multi-Layer Perceptron pot representar funcions no lineals com la XOR.

El perceptró és un model d'un hipotètic sistema nerviós proposat originalment a l'any 1958. Es pot representar amb aquesta forma:



Calcula una suma ponderada de les entrades i les nivela a partir d'una funció d'escala. Això vol dir que el perceptró pot separar l'espai d'entrada en un hiperplà. Normalment això vol dir que nomes es podrien separar problemes lineals.

Rosenblatt va introduir el concepte de pes: nombres que expressen la importància de les diferents entrades a la sortida. Depenent de la sortida de la funció de cada neurona, aquesta generarà un valor 0 o 1 tenint en compte el resultat de si la funció supera o no el valor llindar. La idea principal d'aquest algorisme és aprendre els valors dels pesos que s'apliquen als valors d'entrada per tal de saber si activar una neurona o no.

A més, dintre de l'algorisme existeix una funció d'activació que al mateix temps és una de les parts més importants de les xarxes neurals. Aquestes funcions s'encarreguen d'ensenyar a la neurona en situacions complexes, introdueixen propietats no lineals a la xarxa i a fer petits canvis a la sortida sense tocar els pesos ni les funcions.

Com hem dit abans, el perceptró original no podia solucionar problemes no lineals. Per aquest motiu, es va crear l'algorisme de barckpropagation. Es va proposar l'ús d'una funció diferenciable en comptes d'una funció escalada com a la funció d'activació. Amb aquesta modificació una xarxa amb múltiples nivells era capaç de resoldre aquest tipus de problemes. Exemples de funcions d'activació poden ser: Sigmoidea, Tanh (hiperbòlica), Relu, etc.

2. Descriu l'algorisme Backpropagation proposat per Rumelhart-Hinton-Williams el 1986.

Abans de que es crees l'algorisme de Backpropagation els mètodes d'optimització eren bastant insatisfactoris. El objectiu principal és seleccionar els pesos que ens donen una estimació optima de la funció que modela les dades d'entrenament.

A cada pas de l'algorisme de Backpropagation el que es fa és actualitzar els pesos intentant que aquests s'apropin cada vegada més al seu valor òptim. Aquest valor òptim suposa apropar-se més al valor real de la nostra predicció, dit d'una altra manera, minimitzant l'error per cada neurona de sortida i de tot el conjunt de la xarxa.

El procés de l'algorisme es a la inversa de la direcció de la xarxa. Es comença per la sortida, on s'utilitza la regla de la cadena per tal de calcular derivades i simplificar el nostre càlcul. Després de fer el pas a la sortida de la xarxa, es fa un pas molt similar a les capes ocultes de la xarxa. Utilitzant de nou la regla de la cadena calculem de nou els pesos pertanyents a aquestes capes, amb l'única diferencia que ara en comptes de tenir en compte el valor predit (sortida) i el nostre objectiu, ara tenim en compte que cada sortida de cada capa oculta afecta a la sortida de la xarxa.

Aquest procés es repeteix un nombre bastant alt de vegades per tal de trobar el valor òptim de cada pes dintre de cada neurona de la xarxa minimitzant l'error total.

3. Descriu quines consideracions principals cal tenir en compte a l'hora d'entrenar xarxes neurals

A l'hora d'entrenar xarxes neurals s'han de tenir en compte certs aspectes del model. Els més importants són els paràmetres donats al model per entrenar-se. Aquests paràmetres reben el nom de hiperparàmetres. Aquests hiperparàmetres s'estableixen abans de fer l'entrenament de dades.

Alguns d'aquests hiperparàmetres són:

- **nHidden Layers:** Els hidden layers són el nombre de capes que hi ha entre l'entrada i la sortida. Un nombre molt petit pot causar underfitting.
- **Dropout:** Cancel·la neurones aleatòries per tal de incrementar la precisió a la validació. Té millors resultats quan la xarxa és molt gran. Això fa que el model aprengui de representacions independents.
- **Funció d'activació:** Les funcions d'activació introdueixen elements no lineals als models. Determinen quin valor han de tenir les neurones per tal d'activar-se o no.
- **Learning Rate:** El coeficient d'aprenentatge és un valor fonamental en el procés d'aprenentatge. Un valor molt baix l'algorisme tindrà poca velocitat però convergirà molt suaument mentre que un valor molt alt ens donarà una velocitat bona però hi ha una alta probabilitat que no convergeixi. La millor implementació és aquella on el seu valor va decreixent a mida que es va apropant a la solució òptima.
- **Momentum:** El momentum ens ajuda a saber la direcció del següent pas tenint en compte el coneixement adquirit de les nostres dades. Ajuda a prevenir les oscil·lacions.
- **Nombre d'epochs:** És la quantitat de vegades que el conjunt d'entrenament està exposat al model mentre aquest s'entrena. Un valor massa alt pot provocar overfitting.

Una praxis molt correcta és la de implementar un grid search per tal de trobar els hiperparàmetres òptims per la configuració de dataset que té el nostre problema. Com que molts dels hiperparàmetres estan relacionats, el que fa el grid search és desplaçar lleument els hiperparàmetres per mantenir aquestes relacions però estudiant els canvis que aquests experimenten amb petits canvis.

Bloc 5.1: Memorització

1. Explica com afecta la tria del número de veïns K en l'algorisme K-NN.

L'algorisme de K-NN funciona de la següent manera: Si tenim 2 classes, i diferents valors en el nostre dataset repartits en un pla, la K de l'algorisme ens diu quants veïns tenim en compte respecte l'objecte que volem predir. Si per exemple la K és 3, l'algorisme agafa els 3 veïns més propers i classifica l'objecte a predir com la classe amb més coincidència. És important que aquesta K sigui imparella per a que així mai pugui ocórrer un empat.

Aquesta tria es pot fer manualment seleccionant un valor de K en concret o fent un cross-validation. Independentment del mètode escollit, la funció de cost i la precisió de la predicció canviaran segons el valor K seleccionat. Un valor baix de K resultarà en un model més flexible, amb un baix bias però una variància alta. Amb un valor alt de K en canvi, tindrem més influencia per part dels objectes que ja coneixem i per tant tindrem molt més outliers. Tindrem una regió molt més suau dibuixada tenint una variància molt baixa però un bias més alt.

Una millora que es podria implementar a l'algorisme podria ser un sistema de votació amb pesos. En aquest sistema, el pes relatiu a la votació de cada veí s'estableix amb la distancia entre el veí i l'objecte objectiu. Contra més a prop estigui el veí de l'objectiu més pes tindrà la seva influencia dintre de la votació final del KNN.

Com a curiositat, la elecció de K com el nombre de l'arrel quadrada del nombre de conjunt de dades totals sembla ser la més correcta universalment.

2. Descriu el mecanisme de votació de veïns en els K-NN al fer al regressió.

L'algorisme K-NN funciona tant en el cas d'un model regressor com un model classificador. En el regressor, el K-NN serveix per determinar el valor que obtindrà un objecte en particular tenint en compte els valors d'altres coneguts. Aquests altres s'anomenen veïns, i la seva influencia es clau per tal de seleccionar el valor de la nostra predicció.

El primer pas de l'algorisme és calcular la distància entre el punt a predir i tots els punts d'entrenament. Depenent del valor K que es determina a l'algorisme, s'escullen els k veïns més propers del punt a predir. Dels k veïns més propers, es fa la mitja dels seus valors i se li aplica a la nostra predicció, sumant tots els valors dels veïns i dividint-los per K.

Els mètodes per calcular la distància poden ser: Euclidiana, Manhattan o Hamming.

Per a trobar el millor valor de K es pot fer servir una gràfica tenint en compte l'error d'entrenament i l'error de validació. El punt òptim de K sol ser quan la corba de la gràfica de l'error de validació dibuixa un colze (marca el mínim i torna a pujar).

3. Descriu el mecanisme de votació de veïns en els K-NN al fer classificació.

Al fer classificació, l'objectiu de la predicció canvia respecte un regressor. Anteriorment, al model regressor l'objectiu era obtenir un valor continu tenint en compte les distàncies dels K-veïns més propers. En la classificació en canvi, es busca classificar un objecte en classes, ja sigui binomial (0-1) o multi-classe.

En aquest cas, es calculen les distàncies de tots els punts amb el punt a predir i s'obtenen els K-veïns més propers a ell. Per tal de classificar correctament el nostre punt, s'atribueix al punt a predir la classe més repetida entre els veïns més propers, de manera que es té més a prop a la classe seleccionada.

Bloc 5.2: Clustering

1. Descriu la importància del número de clústers en clustering.

El procés de K-Means clustering es basa en partir les dades en K clústers (regions) de dades on cada clúster pertany a una classe diferent.

Es seleccionen el nombre de centroides que seran els punts centrals dels clústers i es desplacen fins que els centroids estiguin col·locats de manera que separin correctament les classes de dades.

Per a trobar el valor correcte de clústers es pot utilitzar la gràfica d'MSE respecte el nombre de clústers utilitzats. La majoria de gràfiques tenen un element anomenat colze que o atura el ritme de l'error i comença a ser menys rentable tenint en compte el cost computacional.

Normalment el número de clústers óptim per a qualsevol problema sol ser el nombre de classes que hi ha al nostre problema.

També es pot utilitzar el mètode de Silhouette, el qual si que és un mètode algorítmic on es construeixen varis models de clústers amb k-means amb diferent valor de k per avaluar-los i determinar quin és el valor òptim de clústers per aquest problema en particular. Per avaluar cada un dels models es pot utililitzar el mètode de Silhouette, el qual genera un coeficient a partir de la observació i al qual es fa mitjana per obtenir la Silhouette Score.

Aquest coeficient combina la distancia mitjana entre els punts del mateix clúster amb la distancia mitjana del clúster més proper per tal de assignar-hi un valor entre -1 i 1. Un valor negatiu denota que la observació esta probablement errònia, mentre que un valor més proper a 1 denota que la observació és correcta i que encaixa amb el clúster, que a la vegada també esta ben separat dels altres clústers.

Bàsicament, aquest coeficient mesura quant lluny esta la observació dels clústers veïns, on el desitjable és que aquesta distancia sigui la màxima possible, el que ens garanteix una correcta classificació amb el mètode de clústers.

Uns altres mètodes per trobar el nombre òptim de clústers poden ser el gap statistics, hierarchical clustering, calinsky criterion, bayesian expectation maximization i affinity propagation clustering.

2. Descriu com fer clustering amb una mixtura de Gaussianes.

Quan els punts de les diferents classes es sobreposen entre ells, un algorisme tipus K-Means no funcionaria ja que establiria els punts per distància respecte el centroide i classificaria malament alguns punts.

L'objectiu dels models amb mixtura de Gaussianes és trobar els paràmetres dels Gaussians que tinguin una millor representació de les dades.

Aquest procés es fa amb l'algorisme anomenat Expectation-Maximization Algorithm. L'algorisme té dos passos: expectation i maximization. El primer pas és l'expectativa. En aquest pas es calculen les probabilitats de que cada punt s'hagi generat per cada un dels k Gaussians. A diferencia del K-Means, no apliquem una assignació dura aplicant la classe més freqüent, sinó que nomes calculem les probabilitats de cada conjunt de Gaussiana.

La segona part de l'algorisme és la maximització. En aquest procés s'actualitzen els pesos, les mitjanes i les covariàncies. Tal i com es fa en el K-Means, s'agafen les mitges dels punts assignats al clúster per ser la nova mitja, però en aquest cas tindrem en compte també les expectatives calculades en el pas anterior. Per a actualitzar els pesos, es sumen la probabilitat de pertànyer a cada gaussiana i es divideix per el nombre total de punts. Per les mitjanes, es calculen els de tots els punts ponderats per la probabilitat de que aquest punt sigui generat per cada Gaussiana. Per a la covariància, es calcula la de tots els punts ponderades per la probabilitat de que el punt hagi estat generat per la Gaussiana. Es fan tots els càlculs per cada gaussiana.

3. Descriu les diferències entre el K-Means (hard-assignment) i l'EM (soft-assignment).

Mentre el hard-assignment s'assigna directament a una classe la qual sol ser la més freqüent, al soft-assigment es calculen les probabilitats de que el punt pertanyi a les classes per després fer l'altre pas de l'algorisme.

En el cas del K-Means és molt sensible a dades amb soroll o que simplement els punts de les dades estiguin sobreposats. A l'EM en canvi al poder fer desplaçaments direccionals en les zones de cada classe, és menys sensible al soroll i pot ser molt més flexible.

Al l'algorisme de K-Means només està el pas de calcular la distancia i la votació de veïns per tal de catalogar la classe al punt mentre que a l'EM es calculen els pesos, mitjanes i covariàncies a part de calcular la probabilitat de que el punt pertanyi a cada classe.

Mentre que el K-Means intenta minimitzar la diferencia entre els punts de cada clústers (errors quadràtics), els centroides del EM son fets a mida amb clústers més grans i densos.