Resum Parcial 3 Machine Learning

1. A few useful things to know about Machine Learning

Els algoritmes de ML són capaços de fer tasques importants generalitzant exemples. És possible i efectiu en recursos on la programació manual no ho és. Quantes més dades tinguem, més ambiciosos són els problemes que podem solucionar. L'article fica d'exemple els classificadors, que es basen en rebre un vector input discret o continu i donen com a output un valor únic discret.

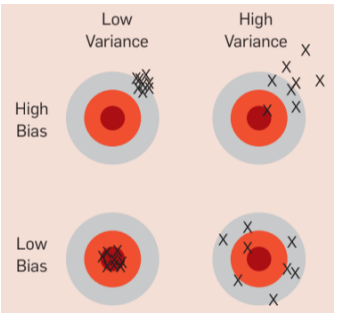
Tots els algoritmes que hi ha avui dia consisteixen en combinacions diferents de 3 components essencials:

- **Representació**: espai d'hipòtesi: sèrie de classificadors en els que pot classificar l'algoritme. Si una classe no està en l'espai d'hipòtesi llavors l'algoritme no ho pot aprendre.
- **Avaluació (objective/scoring function)**: distingir classificadors bons dels dolents. La funció interna i externa poden ser diferents.
- **Optimització**: normalment es comença amb optimitzadors estàndard però amb el temps es van creant personalitzats.

Com la idea és generalitzar dades fora del training set, no tinguis la il·lusió d'èxit sense haver-ho provat amb dades de testing. Per això, quan es fa l'entrenament s'han de guardar dades per a poder fer servir pel testing. Aquestes dades de testing poden corrompre el model, per exemple si es fa servir per a tunejar paràmetres. Això es pot evitar per exemple fent cross-validation, que consisteix en dividir les dades en per exemple 10 porcions i entrenar el model fent servir totes les porcions menys una.

Per a ML a diferència d'altres funcions d'optimització, en aquest cas no tenim la funció que hem d'optimitzar, només ens basem en l'**error d'entrenament**.

Per moltes dades que tinguem mai serà suficient, per anar més enllà hem de fer assumpcions i tenir coneixement previ per a poder generalitzar més enllà del training dataset que tenim. Per sort, en la vida real hi ha patrons que es poden aplicar a ML per a poder fer que els models siguin exitosos, hi ha exemples similars que es poden classificar en la mateixa classe.



Overfitting s'utilitza per a dir que un model ha començat a alucinar trobant particularitats aleatòries als individus de les dades. L'indicador més típic és tenir 100% accuracy al training set i un ~50% al testing. Una manera senzilla d'entendre-ho és dividint l'error en dues coses:

- **Bias**: tendència del model a aprendre la mateixa cosa que està malament.
- **Variància**: tendència del model a aprendre coses aleatòries no respectives a la realitat.

És bastant típic que assumpcions falses fortes siguin millors que assumpcions certes dèbils, perquè un model amb aquesta última necessita més dades per evitar overfitting. Hi ha diversos mètodes per evitar overfitting:

- Afegir el **terme regularitzador (alpha)** a la funció avaluadora penalitzant les branques dels arbres amb estructures més complexes.
- Fer el **test de significat estadístic** per veure si l'estructura de la classe és similar o no.

Tenir dades del training set amb la classe etiquetada incorrecta pot empitjorar el overfitting, però tenir **soroll** en les dades no és la única cosa que pot produir overfitting. També passa quan s'intenta avaluar en el testing diferents hipòtesis i el model està basat en tests estadístics estàndards que només avaluen una hipòtesi.

False discovery rate: controlar el nombre d'hipòtesis no nul·les acceptades erròniament.

Curse of dimensionality: com més paràmetres/features vulguem tenir en compte, més difícil és classificar per al model. Models com el k-nearest neighbor deixen de funcionar perquè hi ha masses paràmetres a tenir en compte i massa soroll, a un cert punt tots els punts semblen estar a prop en 2 dimensions quan l'espai sencer són d dimensions.

Blessing of non-uniformity: normalment els exemples no es reparteixen en les diferents dimensions, sinó que normalment els que són similars solen estar a prop entre ells o estan tots en les mateixes dimensions menors.

No hi ha manera de garantitzar quan encertarà o fallarà un model, ni tampoc de determinar quan és bo o dolent a partir de cert ràtio de fallades. La theoretical guarantees només serveix per entendre el disseny de l'algoritme.

La major part del temps per fer un model no es gasta en fer ML sinó en preprocessar les dades, ajustar paràmetres, analitzar resultats, etc. Normalment que el model aprengui sol ser el més ràpid de tots els processos. Feature engineering és difícil perquè és domain-specific, però la idea és que els ML puguin poc a poc començar a automatitzar això.

Normalment un model amb més dades que un altre més intel·ligent però amb una quantitat de dades modesta acabarà sent millor per la quantitat de dades. Però les dades no és el més important ara (a diferència de fa 20 anys), sinó el temps. Tenir un model massa complex que tarda massa en aprendre no és factible avui dia, de manera que s'acaben fent servir més de simples. Hi ha dos tipus de models:

- **Tamany fix**: la seva representació té un tamany fix, com classificadors lineals.
- **Tamany variable (non-parametric learners)**: la seva representació canvia en funció de la quantitat de dades que tenim, per exemple arbres de decisió.

Avui dia no hi ha algoritme preferit, sinó una combinació de tots és el que fa que sigui millor.

- **Bagging** es refereix a crear diferents versions del training set i combinem els resultats votant. Redueix variància i incrementa només un pèl bias.
- **Boosting:** els exemples del training set tenen pesos, de manera que en noves iteracions es tingui més en compte els errors anteriors.

- **Stacking:** l'output de classificadors individuals es converteixen en inputs d'un model més alt nivell que mira una manera de combinar-los.

La diferència entre model ensembles i Bayesian model averaging (BMA) és que en el primer tenim pesos més o menys similars i en el bayesian ficar un pes és tan gran que és gairebé com escollir un algoritme.

Occam's razor diu que si tenim dos classificadors amb el mateix error d'entrenament, el més simple dels dos ho farà millor en les dades de testing. Però això no sempre és cert, l'exemple més bàsic és les màquines vectorials de suport que no tenen overfitting per molts paràmetres que els hi fiquis.

Només perquè una funció sigui representable no vol dir que es pugui aprendre.

Correlació no implica causalitat. L'exemple de que bolquers i cervesa es compren junts diu que hi ha una correlació entre les variables, si els posem junts vendrem més que ficant-los separats. Però això és només una suposició, ja que el que tenim són dades observades, mentre que el model normalment té dades experimentals on el model no ha pogut influir en la decisió.

2. The Barcelona declaration for the proper development and usage of artificial intelligence in Europe

La IA s'ha aplicat en molts camps, inclòs no tecnològics. Mentre que la gent es preocupa per si les IAs seran capaces de reconfigurar-se a si mateixes i acabar dominant el món, els que treballen en l'àmbit saben que no és possible que passi i tenen altres preocupacions:

- Es fa més difícil lidiar amb la societat perquè les xarxes socials polaritzen les opinions. Per a poder fer accions col·lectives es necessita estar d'acord i en aquest cas la IA hauria de facilitar un consens, no destruir-lo.
- Deep learning és el mètode que ha provat ser més efectiu per reconèixer veu però temes humanístics com decisions financeres o aplicació de la llei són problemàtics des del punt de vista humanístic. També gent que busca feina està frustrada perquè no sap com passar el filtre de la IA, la qual es basa en la classe, raça, gènere, etc.
- Altres temes que fan pensar és qui serà el responsable en cas d'accident en els cotxes autònoms? Com encara no hi ha cap consensus per evitar el seu desenvolupament, no s'ha pres cap mesura fins al moment.

A Barcelona s'han fet inicitatives per a evitar el desenvolupament desenfrenat de les IAs. Es va fer un workshop:

- Sessió 1: està la IA preparada per a ser desplegada a gran escala? Està el State of the Art preparat per a afrontar els reptes que demana la societat? Com poder-la fer confiable?
- Sessió 2: com es pot fer servir per a fer el bé, no només per a l'ús comercial? Com afecta la propaganda política a la IA i com podem tractar els seus problemes?

També s'han fet presentacions:

- Sessió 1: com està presentada la IA en la cultura popular?

- Sessió 2:
    - Part A: com transforma l'accés a la informació.
    - Part B: per què ha crescut tant i tan ràpidament?
- Sessió 3: millors pràctiques per fer desenvolupament i desplegament de IA.

Es va signar una declaració pels participants d'aquests debats. Resum:

1. **Scope**: considerem:
    a. Knowledge-based IA: intenta modelar el coneixement humà en termes computacionals. Fa servir ontologies, sentit comú general, etc.
    b. Data-driven IA: deep learning per exemple que en base a una gran quantitat de dades es busquen patrons per a poder predir el comportament en una futura situació.
2. **Inversió**: Europa necessita ficar més esforç en IA.
3. **Prudència**: es necessita comunicació honesta sobre les fortaleses i límits de les aplicacions amb IA.
4. **Fiabilitat**: és necessari crear una organització per verificar i validar la IA.
5. **Responsabilitat**: les aplicacions amb IA han de ser intel·ligibles, poder explicar la base de les seves decisions. Responsabilitat ha de venir abans que el desplegament de la IA.
6. **Identitat**: sempre ha de ser clar si estem tractant amb una IA o amb un humà. La solució pot ser obligar a ficar una marca d'aigua per a trobar els culpables en cas de que la IA s'hagi utilitzat per a propòsits com manipular les opinions de la gent, extorionar, etc.
7. **Autonomia**: es necessita ficar regles per restringir el comportament autònom.
8. Mantenir coneixement humà: la intel·ligència humana s'ha de preservar com a coneixement futur.

Problemes que van sorgir durant l'establiment de la declaració:

1. **Avui dia hi ha una necessitat encara més gran de clarificar què volem dir amb IA quan discutim temes legals i ètics.** Es creu que la paraula IA s'està fent servir de paraigües per a referir-se a una gran quantitat d'operacions com són el reconeixement de patrons, anàlisi estadístic, etc.
2. **Els plans per a suportar el desenvolupament i desplegament de IA encara s'han de concretar a Europa.** Hi ha plans per a fer coses però a Europa és força raro trobar finançament.
3. **Necessitem més prudència amb el que la IA pot fer.** Els rumors de que la IA ens conquerirà només són rumors, no són possibles realment. Una IA no pot intentar enganyar i no pot saber quan l'altre està enganyant. Posa l'exemple de que l'article a Facebook dient que van parar un experiment perquè dues IAs van començar a parlar un llenguatge que nosaltres no enteníem, però que realment no el van parar per això sinó que jugaven amb les paraules i confonien al públic, quan en cap moment es deia explícitament que s'havia aturat l'experiment per això.
4. **No hi ha mecanismes de certificació de les IAs.** És difícil fer una certificació per a les IAs per a determinar quins requeriments ficar.
5. **Responsabilitat a través de IA explicable i persona legal.** Es necessita poder explicar com una IA ha arribat a una decisió concreta i qui és el responsable

d'aquesta decisió si algo va malament. D'aquesta manera podem veure com millorar-la i si té algun bias.

6. **La identitat de sistemes de IA segueix estant borrosa.** La persona ha de poder saber que està parlant amb una IA, la meta de la IA no ha de ser poder enganyar als usuaris perquè pensin que estan parlant amb una persona. A les persones no els importa estar parlant amb una IA si saben que és una IA.
7. **La major part dels problemes de l'autonomia de la IA encara estan oberts.** El problema és veure qui és el responsable quan es prenen decisions com IAs amb armes, o amb cotxes autònoms, etc. Per als cotxes autònoms a Alemanya s'ha implantat una llei que et fa responsable legalment i que els desenvolupadors han de seguir. Com a Europa no s'ha implantat cap encara, és possible que en un futur els requeriments col·lisionin amb els d'altres països. Hi ha una segona manera d'abordar-ho que és aplicant moralitat i ètica als models, però això encara està molt verd.
8. **Mantenir coneixement humà.** Les IAs no s'han de fer servir per a reemplaçar els humans sinó tractar-les com a eines i assistents que ens ajuden a ser més eficients. Cadascun tenim diferents virtuts: nosaltres ens adaptem millor als canvis i a les coses noves, mentre que les IAs poden recordar moltíssima informació i trobar patrons.

La conclusió és que s'ha d'educar en aquest tema per a no tenir backlash d'expectatives no complertes i per a evitar el mal ús de la IA i efectes secundaris. A més, la responsabilitat final del seu ús acaba sent tant dels usuaris com dels seus desenvolupadors en cas d'accions malicioses.

3. Resum temari classe

3.1 Introducció general a machine learning (ML)

L'aprenentatge automàtic s'analitza com una disciplina destinada a desenvolupar algoritmes capaços de realitzar tasques complexes mitjançant la identificació de patrons en dades.

Un aspecte important a considerar en l'aprenentatge supervisat és l'error inherent associat a les dades etiquetades. Aquest error pot derivar tant de biaixos humans en el procés d'etiquetatge com de la manca de precisió en la definició de les categories. Per exemple, en problemes complexos com la classificació d'espècies biològiques, les diferències poden ser subtils i subjectives, reflectint la limitada comprensió humana de la realitat. Aquests errors poden impactar en la capacitat del model per generalitzar i predir amb precisió, fins i tot quan l'algorisme en si funciona de manera òptima. Per tant, és fonamental tenir present que els models d'aprenentatge supervisat només poden ser tan fiables com ho són les dades amb què s'han entrenat.

Exemples:

Reconeixement de dígits manuscrits:

- El sistema processa imatges de dígits i els classifica com a nombres del 0 al 9.
- El desafiament principal rau en les variacions en els estils d'escriptura dels usuaris.
- Per exemple, com el "6" pot tenir diferents formes segons el manuscrit.
- La solució es basa en entrenar models que puguin generalitzar i identificar característiques comuns malgrat aquestes variacions.

Conjunt de dades Iris:

- Es classifiquen flors segons atributs com llargada i amplada dels pètals i sèpals.
- L'objectiu és predir l'espècie d'una flor nova basant-se en exemples etiquetats.
- Aquest problema és un clàssic de la classificació supervisada.

Categories de problemes en ML:

- **Classificació:** Assignació d'exemples a categories predefinides.
- **Regressió:** Predicció de valors continus (com preus d'habitatges).
- **Altres tasques:** Exploració de patrons o agrupacions sense etiquetes (clustering).

Avantatges de ML:

Automatització de tasques complexes: L'aprenentatge automàtic permet resoldre problemes que són difícils de programar manualment, com el reconeixement d'imatges o la classificació de dades, amb menys esforç de programació.

Capacitat d'adaptació: Els models poden ajustar-se a dades canviants, aprendre patrons nous i millorar amb el temps amb més dades d'entrenament.

Diversitat d'aplicacions: El ML té aplicacions en camps molt variats, com la biologia computacional, les finances, la detecció de fraus, la medicina, la classificació automàtica de documents i la predicció del comportament del mercat.

Generalització: Els bons models de ML són capaços de fer prediccions fiables en dades noves, sempre que estiguin ben entrenats i no sobreajustats.

Optimització de recursos: A través de l'ús d'algorismes d'ML, es pot aconseguir una millor presa de decisions basada en dades i reduir la necessitat d'intervenció humana en certes tasques.

Inconvenients ML:

Sobreajustament: Un problema comú és que els models poden ajustar-se massa a les dades d'entrenament, fallant en generalitzar bé a dades noves. Això pot succeir amb models excessivament complexos.

Dependència de les dades: La qualitat dels resultats d'ML depèn de la qualitat de les dades. Si les dades estan incompletes, són sorolloses o tenen biaixos, això es reflectirà en els resultats dels models.

Complexitat tècnica: La implementació i optimització de models d'aprenentatge automàtic pot requerir coneixements especialitzats i una comprensió profunda del domini d'aplicació.

Alt cost computacional: Alguns mètodes avançats, com les xarxes neuronals profundes, poden requerir un gran poder computacional i temps d'entrenament prolongat, especialment amb grans conjunts de dades.

Problemes ètics i de privacitat: Hi ha riscos associats amb l'ús de dades personals, com en els sistemes de reconeixement facial o la detecció de fraus, que poden plantejar dilemes ètics i problemes de privacitat.

3.2 Tipus d’aprenentatge automàtic (AA)

3.2.1 Aprenentatge supervisat

Quan s’utilitza:

- Quan tenim un conjunt de **dades etiquetades** amb **resultats coneguts**.
- Requereix una **associació clara** entre les característiques d’entrada (**atributs**) i els valors objectiu (**etiquetes**).
- Es fa **l’assumpció** que el **conjunt d’entrenament** és **representatiu del món real** (les dades futures són similars a les d’entrenament).

Objectiu:

- Entrenar un **model que pugi predir etiquetes** o valors desconeguts per a noves dades.
- En classificació: assignar una instància a una classe concreta.
- En regressió: estimar un valor numèric.

Problemes comuns:

- **Sobreajustament** (overfitting):
    - **Problema**: Quan el model s'adapta massa a les dades d'entrenament i perd capacitat de generalitzar.
    - **Resultat**: Mal rendiment en dades noves (no vistes en l'entrenament).
    - **Solució**: optimitzar el grau del model. Divisió de dades.
- **Subajustament** (underfitting): Quan el model és massa simple i no captura les relacions entre les dades.
- **Dades desequilibrades**: Quan una classe domina en el conjunt d'entrenament, el model pot esdevenir esbiaixat.

Importància de la Representativitat del Conjunt d'Entrenament

En l'aprenentatge supervisat, assegurar que el conjunt d'entrenament sigui representatiu del món real és fonamental per garantir la generalització del model. Si les dades d'entrenament no reflecteixen adequadament la distribució de les dades reals, el model pot mostrar biaixos significatius i un baix rendiment en situacions pràctiques. Això pot ocórrer, per exemple, quan hi ha desequilibris de classes o quan certes condicions del problema no estan ben capturades en el conjunt inicial.

Aquesta representativitat es pot millorar utilitzant tècniques com la selecció estratègica de dades, l'augment de dades per a classes minoritàries o la validació amb múltiples conjunts per detectar possibles problemes de biaix. A més, cal tenir en compte que els errors humans en l'etiquetatge també poden influir en els resultats, afegint un grau de soroll que pot desviar el model de la realitat.

Exemples d'aplicacions:

- **Classificació**: Identificar correus com a spam o no-spam, reconeixement facial.
- **Regressió**: Predir preus d'habitatges, preveure consums energètics o valors borsaris.

3.2.2 No supervisat

Quan s'utilitza:

- Quan **no es disposa d'etiquetes** per a les dades.
- Les **dades** han de tenir **estructures o patrons latents** que es poden descobrir.
- Es fa l'assumpció que les **dades es poden agrupar o estructurar** d'alguna manera útil (clústers, dimensions reduïdes, etc…)

Objectiu:

- **Descobrir** estructures ocultes, agrupaments o patrons en les dades.
- **Aconseguir representacions simplificades** de dades complexes.

Problemes comuns:

- **Interpretabilitat**: Els resultats poden ser difícils d'interpretar, especialment en clustering.
- **Sobregeneralització**: Algoritmes que identifiquen patrons allà on no n'hi ha.
- **Escalabilitat**: Treballar amb dades molt més grans pot ser computacionalment costós.

Definició de Semblances i Jerarquia en Agrupacions

Una de les dificultats principals en l'aprenentatge no supervisat és definir adequadament la semblança entre dades per formar agrupacions significatives. La qualitat dels clústers depèn de la mesura de semblança escollida, com ara la distància euclidiana, la distància cosinus o altres mètriques específiques que capturen millor la naturalesa de les dades.

A més, moltes dades complexes presenten estructures jeràrquiques, on els grups no són independents sinó que estan organitzats en diferents nivells de granularitat. Per exemple, en una estructura jeràrquica com un dendograma, cal decidir a quin nivell "tallar" per obtenir els clústers finals, fet que requereix una comprensió prèvia de la naturalesa de les dades. Aquest enfocament és especialment útil per problemes amb una complexitat intrínseca elevada, ja que pot capturar les relacions entre grups de manera més detallada.

Exemples d'aplicacions:

- **Clustering**: Segmentació de clients, agrupament d'usuaris en xarxes socials.
- **Reducció de dimensionalitat**: Visualització de dades d'alta dimensió.
- **Regles d'associació**: Anàlisi de cistelles de compra, detecció de patrons de consum.

3.2.3 Per reforç

Quan s'utilitza:

- S'utilitza quan un agent pot interactuar amb un entorn.
- Requereix un **mecanisme de recompenses** que indiqui al model quan les accions són correctes o incorrectes.
- Es fa l'**assumpció que l'entorn proporciona informació suficient** per aprendre estratègies òptimes.

Objectiu:

- Aprendre una **estratègia òptima** que **maximitzi** les **recompenses** acumulades.
- Desenvolupar **sistemes autònoms** capaços de prendre **decisions seqüencials**.

Problemes comuns:

- **Exploració vs explotació**: **Determinar quan explorar** noves accions **o explotar coneixement** adquirit.
- **Sparse rewards**: Situacions on les **recompenses són escasses o retardades**, fent l'**aprenentatge més difícil**.
- **Alt cost d'error**: En certs entrons, els errors poden ser costosos de corregir.

Exemples d'aplicacions:

- **Robòtica**: Robots que aprenen a caminar o a manipular objectes.
- **Jocs**: Algoritmes que juguen a videojocs o escacs millor que els humans.
- **Gestió de recursos**: Optimització de trànsit, control d'inventaris.

3.3 Mètodes i algorismes

3.3.1 Aprenentatge Supervisat:

3.3.1.1 K-Nearest Neighbors (KNN) - Classificació

Descripció bàsica del mètode:

- KNN és un algorisme d'aprenentatge supervisat utilitzat per a classificació i regressió.
- La idea principal és assignar a una instància la classe més freqüent entre els seus **K veïns més propers** (en termes de distància).
- Quan K=1K = 1K=1, la classificació es fa segons la classe del veí més proper. Quan K>1K > 1K>1, es prenen els veïns més propers i s'aplica una votació per majoria per decidir la classe.

Ponderació per distància en KNN

Un problema comú en l'ús de KNN és la sensibilitat a dades desequilibrades i sorolloses. Per abordar-ho, s'utilitza la **ponderació per distància**, que assigna més pes als veïns més propers durant la predicció.

- **Mecanisme**: Cada veí contribueix a la classificació o regressió en funció d'un pes proporcional a la seva distància. Per exemple, el pes pot ser invers a la distància: $w_i = \frac{1}{d_i}$, on d_i és la distància al veí i.
- **Avantatges**:
  - Redueix l'impacte del soroll de veïns més llunyans.
  - Millora l'eficàcia en dades desequilibrades, ja que dóna més importància als punts propers a l'exemple objectiu.
- **Aplicació**: És especialment útil en conjunts de dades amb distribucions no uniformes o on algunes classes són minoritàries.

Funcionament

1. **Càlcul de distàncies**: Les distàncies entre la instància objectiu i totes les instàncies del conjunt d'entrenament es calculen mitjançant una mètrica, com ara:
2. **Selecció dels veïns**: Es seleccionen els **k veïns més propers** basant-se en la mètrica de distància.
3. **Predicció**:
   - En **classificació**, la classe es decideix per **votació majoritària** entre els veïns seleccionats.

    - En **regressió**, s'utilitza la **mitjana** dels valors dels veïns, ponderada segons la distància si es desitja.

Càlcul de la distància:

Distància Euclidiana:

- És el mètode més habitual. Es calcula com la distància recta entre dos punts en un espai n-dimensional.
- S'utilitza en situacions on les característiques tenen magnituds comparables.
- Formula: $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$.

Distància de Manhattan:

- També coneguda com a distància de la ciutat o taxicab, es calcula com la suma de les diferències absolutes entre les coordenades corresponents.
- S'utilitza en espais on el moviment segueix una graella, com en xarxes de carreteres.
- Formula: $\sum_{i=1}^{n} |x_i - y_i|$.

Distància de Minkowski:

- És una generalització de les distàncies Euclidiana i de Manhattan.
- El paràmetre p determina la seva forma. Quan p=2p = 2p=2, és la distància Euclidiana, i quan p=1p = 1p=1, és la distància de Manhattan.
- Formula: $(\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$.

Distància Cosinus:

- Més adequada per dades de text o vectors de freqüència. Mesura la diferència angular entre dos vectors.
- Formula: $1 - \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \|\vec{y}\|}$.

Distància de Mahalanobis:

- Té en compte la correlació entre les variables i és útil en espais amb escales de magnituds molt diferents.
- Formula: $\sqrt{(x - y)^T S^{-1} (x - y)}$, on S és la matriu de covariances.
-

Selecció del valor de k

- **k petit**:
    - Major adaptabilitat a exemples específics.
    - Risc de sobreajustament (**overfitting**), ja que el model es veu afectat pel soroll.

- **k gran**:
    - Reducció de la influència del soroll.
    - Possible subajustament (**underfitting**), ja que es perden patrons locals.

L'elecció òptima de k es pot determinar mitjançant tècniques com **validació creuada**. Mes a la secció 3.11.

Tècnica del "hold-out":

- Mantenir un conjunt de dades no utilitzat durant l'entrenament, conegut com a conjunt de test.
- Un cop seleccionat K , verificar-ne el rendiment sobre aquestes dades mai vistes.

Impacte de la normalització

Les característiques amb escales diferents poden desviar el càlcul de distàncies. Per això, normalitzar les dades (e.g., escalar a un rang [0,1]) és fonamental en KNN.

Ponderació per distància:

- Assignar pesos als veïns segons la seva distància a l'exemple diana, independentment de K.
- Això ajuda a optimitzar el rendiment quan la selecció exacta de K no és clara.

Aplicacions:

- Classificació: ús típic per assignar etiquetes categòriques a instàncies desconegudes.
- Regressió: pot predir un valor numèric continu basant-se en la mitjana ponderada (o no ponderada) dels valors dels veïns propers.
- Exemples mencionats inclouen la classificació d'espècies de flors i la predicció del preu d'habitatges.

Aspectes tècnics:

- **Ponderació per distància**: En problemes de regressió, es pot aplicar una ponderació inversa a la distància, donant més pes als veïns més propers.
- **Sobreajustament**: S'esmenta que KNN és susceptible a dades sorolloses o exemples atípics, per això es poden eliminar instàncies que siguin discordants en l'espai de característiques. Mes a la secció 3.12.

Avantatges i limitacions:

- **Avantatges**:
    - Simplicitat d'implementació i intuïtiu d'entendre.
    - Eficaç per a problemes amb estructures locals complexes.
- **Limitacions**:
    - La complexitat computacional pot ser elevada per conjunts de dades grans, ja que cada predicció requereix calcular distàncies a totes les instàncies d'entrenament.

- Sensible a la qualitat del conjunt de dades i la normalització de les característiques.

3.3.1.2 Arbres de decisió - Classificació

1. Conceptes bàsics

- **Definició**: Un arbre de decisió és un model predictiu que utilitza una estructura jeràrquica d'arbres per prendre decisions basades en atributs d'un conjunt de dades.
- **Funcionament**: Els nodes interns representen atributs o característiques de les dades. Les branques representen condicions o regles basades en aquests atributs, i les fulles representen les classes o resultats finals.

2. Construcció

Mètode 1. Construcció de l'arbre de l'arrel a les fulles (incremental)

- **Idea principal**:
  - Es comença amb tota la base de dades al node inicial (arrel).
  - Es selecciona successivament l'atribut més informatiu per dividir les dades en subconjunts més homogenis, basant-se en un criteri com la reducció de la **entropia** o el **gain information**.
  - El procés continua recursivament per a cada subconjunt fins que es compleix un criteri de parada (per exemple, no hi ha més atributs per dividir o totes les instàncies d'un subconjunt pertanyen a la mateixa classe).
- **Avantatges**:
  - És simple d'entendre i implementar.
  - L'arbre reflecteix directament la distribució de les dades.
- **Limitacions**:
  - Pot generar arbres profunds i complexos que poden sobreajustar-se a les dades d'entrenament.
  - És susceptible a dades sorolloses o amb atributs irrellevants.

Mètode 2: Construcció completa seguida de poda

- **Construcció completa inicial**:
  - Es construeix un arbre "complet", és a dir, s'aprofiten tots els atributs i condicions possibles fins que no es pugui dividir més.
  - Això generalment condueix a un arbre molt complex i profund que probablement sobreajustarà les dades d'entrenament.
- **Procés de poda**:
  - Es revisen les branques de l'arbre per eliminar nodes que no contribueixen significativament a la generalització del model.
  - S'avaluen diferents mètodes de poda:
    - **Poda per error de validació**: Retirar branques que no milloren el rendiment en un conjunt de dades de validació.

        - **Poda per complexitat**: Es penalitza la mida de l'arbre, preferint models més simples que mantenen un rendiment acceptable.
        - **Poda basada en mesures estadístiques**: Per exemple, eliminar nodes amb distribucions estadísticament poc significatives.
- **Avantatges**:
    - Millora la generalització, reduint el risc de sobreajustament.
    - L'arbre resultant és més compacte i interpretable.
- **Limitacions**:
    - Requereix més càlculs inicials i necessita conjunts de validació separats.
    - El procés pot perdre informació útil si no es fa adequadament.
- **El procés pot perdre informació útil si no es fa adequadament.**

3. Criteris per escollir els atributs i mesurar la qualitat de la divisió

1. **Entropia i informació**:
    - Es calcula la reducció de l'entropia després d'una divisió.
    - Un atribut amb alta reducció d'entropia s'utilitza per dividir les dades.
2. **Índex Gini**:
    - Una mesura de puresa en els subconjunts resultants d'una divisió.
    - Es prefereixen divisions que resulten en subconjunts més homogenis.
3. **Proporció d'errors**:
    - Es compara el nombre de classificacions incorrectes abans i després de la divisió.

4. Casos especials de construcció d'arbres

- **Atributs numèrics**:
    - Es defineixen llindars per convertir atributs continus en decisions binàries (per exemple, "és menor que X?").
- **Atributs categòrics amb múltiples valors**:
    - Es poden transformar en diverses preguntes binàries o fer una decisió multi-valor directament.

3.3.1.3 Regresió lieal - Regressio

Definició i Context:

- La regressió lineal es presenta com un model senzill que busca ajustar una línia recta als punts de dades per fer prediccions. Aquesta línia representa la relació entre una variable dependent (valor objectiu) i una o més variables independents (atributs).

Casos d'ús:

- Predicció de **valors continus**, com el preu d'un habitatge basat en la superfície construïda.
- Comparació amb models més complexos (xarxes neuronals o polinòmics), destacant la seva simplicitat i eficàcia en situacions amb poques variables o dades limitades.

Avantatges del Model:

- Facilitat d'implementació i ajust.
- Eina de referència per comparar amb altres models més complexos.
- És especialment útil quan un model més simple pot fer prediccions igualment efectives amb menys recursos computacionals.

Aspectes Matemàtics:

- La regressió lineal calcula els coeficients que defineixen la línia (pendent i intersecció) per minimitzar l'error entre les prediccions i els valors reals.
- Es poden aplicar extensions per fer-la més robusta en cas de dades no lineals.

Limitacions i Relació amb Altres Mètodes:

- Es destaca que un model lineal pot no ser suficient per dades molt complexes. Això porta a la necessitat de models polinòmics o no lineals en aquests casos.
- S'utilitza com a punt de partida o model base per demostrar conceptes, abans de passar a tècniques més avançades.

Problemes d'Overfitting:

- En un context d'aprenentatge supervisat, es ressalta la importància d'evitar el sobreajustament fins i tot en models lineals simples. Això es relaciona amb assegurar que les prediccions generalitzin bé en dades no vistes. Mes a la secció 3.12.

3.3.1.4 Adaptacions de KNN per a predir valors continus - Regressio.

Context i idea bàsica:

- KNN es pot aplicar no només per a **classificació**, sinó també per a **regressió**, quan l'objectiu és predir un valor continu en lloc d'una etiqueta categòrica.
- En el cas de regressió, es fa una predicció sobre el valor d'un atribut per a un objecte desconegut utilitzant els valors dels veïns més propers en el conjunt de dades d'entrenament.

Funcionament per a valors continus:

- **Predicció basada en la mitjana ponderada dels veïns:**
  - Per predir el valor d'un atribut, es fa servir la **mitjana dels valors** dels veïns més propers (K veïns).
  - Els veïns contribueixen de manera diferenciada segons la **distància**: com més a prop estiguin, més pes tenen en el càlcul de la predicció.
  - El pes s'estableix com una **funció inversa de la distància**. Per exemple, el pes per a cada veí pot ser proporcional a $\frac{1}{d^2}$, on d és la distància entre l'objecte a predir i el veí.

Exemple matemàtic:

- Es dona la fórmula general:

$$\hat{y} = \frac{\sum_{i=1}^{k} w_i \cdot y_i}{\sum_{i=1}^{k} w_i}$$

  - $\hat{y}$: Valor predit.
  - y_i: Valor del veí iii.
  - w_i: Pes assignat al veí iii, que depèn de la distància (inversament proporcional).
  - k: Nombre de veïns considerats.

Avantatges i consideracions:

- L'ús d'un **sistema de pesos** permet que els veïns més propers influeixin més en la predicció, reduint l'impacte dels veïns més llunyans.
- L'elecció del valor de k és crucial:
  - Un k molt petit pot conduir a un model sensible al soroll (sobreajustament).
  - Un k molt gran pot diluir la influència dels veïns més propers, reduint la precisió.

Aplicacions pràctiques:

- Predicció de valors en propietats físiques, com el preu d'habitatges o mesures contínues basades en característiques similars a les dels veïns més propers.

3.3.2 Aprenentatge No Supervisat: Clustering:

3.3.2.1 Tipus de clustering

Clustering Jeràrquic

- Organitza els punts en una estructura jeràrquica, representada normalment com un **dendrograma**.
- Aquesta tècnica és especialment útil quan volem comprendre les relacions entre els clústers en diferents nivells d'agrupament.

Subtipus:

- **Aglomeratiu (Bottom-Up)**:
  - Comença tractant cada punt com un clúster independent.
  - Progressivament, combina els clústers més propers fins que tots els punts estan en un únic clúster o fins a un llindar especificat.
  - Exemple de distàncies utilitzades per decidir la fusió de clústers:
    - **Distància mínima**: Agrupa els dos punts més propers de dos clústers.

        - **Distància màxima**: Agrupa els dos punts més allunyats.
        - **Distància mitjana**: Considera la mitjana de les distàncies entre tots els punts dels dos clústers.
        - **Centroidal**: Usa el centroid dels clústers com a referència.
- **Divisiu (Top-Down)**:
    - Comença amb tots els punts agrupats en un sol clúster.
    - Successivament, divideix el clúster en subclústers fins a aconseguir el nivell desitjat.

Interpretació de dendogrames

Els dendogrames són eines visuals per interpretar els resultats del clustering jeràrquic.

- Com llegir un dendograma:
    - Cada node representa un clúster.
    - Les altures dels enllaços entre nodes indiquen la distància o la semblança entre els clústers.
- Concepte de tallar el dendograma:
    - Es selecciona un nivell d'alçada per determinar quins clústers finals es formen. Un tall més baix produeix més clústers; un tall més alt, menys.

Avantatges:

- No requereix especificar prèviament el nombre de clústers.
- Proporciona una visió global de les relacions jeràrquiques entre punts.

Inconvenients:

- Alt cost computacional (sobretot en conjunts de dades grans).
- Decidir on "tallar" el dendrograma per obtenir els clústers finals pot ser subjectiu.

Clustering Particional

- Assigna els punts a un nombre fix de clústers prèviament determinat.
- Els clústers es formen optimitzant una funció objectiu, com minimitzar la suma de distàncies quadrades entre punts i el centroid corresponent.

Exemple típic:

- **K-Means**:
    - Divideix els punts en **K clústers**, amb cadascun centrat en un **centroid**.
    - Iterativament:
        - Assigna cada punt al clúster amb el centroid més proper.
        - Recalcula els centroids basant-se en la mitjana dels punts assignats a cada clúster.

Limitacions de K-Means i solucions alternatives

El mètode K-Means presenta algunes limitacions inherents al seu enfocament.

- **Limitacions**:
    - **Forma dels clústers**: Assumeix que els clústers són esfèrics. Això pot fallar en identificar formes irregulars o densitats variables.
    - **Nombre fix de clústers**: Requereix especificar kkk a priori, fet que pot ser problemàtic sense coneixement previ.
    - **Cost computacional**: Per conjunts grans, calcular distàncies en cada iteració pot ser costós.
- **Solucions**:
    - **Mixtures gaussianes (GMM)**: Aprofiten una combinació de distribucions probabilístiques per modelar clústers amb formes arbitràries.
    - **Mètodes jeràrquics**: Permeten explorar la relació entre clústers a diferents nivells sense un kkk fix.

Característiques:

- Basat en distàncies euclidianes, el que significa que funciona millor amb dades normalitzades i esferoides.
- Pot fallar en detectar clústers amb formes irregulars o densitats variables.

Altres mètodes particionals:

- **K-Medoids**:
    - Similars a K-Means, però usa punts reals com a centroids en lloc de la mitjana.
- **Clustering difús (Fuzzy Clustering)**:
    - Assigna cada punt a múltiples clústers amb un grau de pertinença (valors entre 0 i 1).

Clustering Basat en Densitat

- Identifica grups de punts que estan densament empaquetats en l'espai, separats per zones de baixa densitat.

Exemple:

- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**:
    - Defineix clústers com a regions amb una densitat mínima de punts.
    - Té en compte dos paràmetres clau:
        - **Epsilon (ε)**: Radi d'una regió al voltant d'un punt.
        - **MinPts**: Nombre mínim de punts necessaris dins d'aquesta regió per formar un clúster.
    - Classifica punts com a:
        - **Core points**: Punts dins de regions d'alta densitat.
        - **Border points**: Punts a la frontera dels clústers.
        - **Outliers**: Punts que no compleixen els requisits de densitat.

Avantatges:

- No requereix especificar el nombre de clústers.

- Pot gestionar clústers de forma arbitrària i detectar outliers.

Limitacions:

- Sensible a la selecció dels paràmetres ε i MinPts.
- Dificultat per identificar clústers amb densitats molt variables.

Clustering Basat en Models

- Assumeix que les dades es generen a partir d'una combinació de distribucions estadístiques i ajusta un model a les dades.

Exemple:

- **Mixtures Gaussianes (GMM)**:
  - Suposa que cada clúster segueix una distribució gaussiana.
  - Cada punt té una probabilitat d'estar en cada clúster, en lloc d'una assignació dura.
  - Optimitza els paràmetres de la distribució mitjançant l'algoritme Expectation-Maximization (EM).

Avantatges:

- Capacitat per modelar clústers de forma el·líptica o arbitrària.
- Proporciona probabilitats d'assignació, útils en casos amb incertesa.

Limitacions:

- Requereix especificar el nombre de components (clústers).
- Pot ser computacionalment intensiu en conjunts de dades grans.

3.3.2.2 Mètriques i validació de clustering

Cohesió (Intra-cluster Distance)

- Mesura com de compactes són els punts dins d'un mateix clúster.
- La cohesió es calcula sovint com: $$\text{Cohesió} = \frac{1}{N}\sum_{i=1}^{N} \text{dist}(x_i, c)$$
- On x_i són els punts del clúster i c és el centroide del clúster.

Separació (Inter-cluster Distance)

- Avalua la distància entre els diferents clústers, idealment volent-se màxima.
- Una separació elevada significa que els grups són ben definits.

Índex Silhouette

- Combina cohesió i separació en una sola mètrica.

- Per a un punt iii: $$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$
    - a(i): la distància mitjana entre el punt iii i tots els punts del mateix clúster.
    - b(i): la distància mitjana entre el punt iii i tots els punts del clúster més proper.
    - El valor de S(i) oscil·la entre -1 (mala assignació) i 1 (assignació òptima).

Davies-Bouldin Index:

- Mesura la compactació i separació.
- Relació entre la dispersió dins del clúster i la distància entre centres de clústers.
- Menors valors són millors.

Calinski-Harabasz Index:

- Ratio entre la dispersió entre clústers i dins dels clústers.
- Major puntuació indica millor separació.

3.4 Metodología

3.4.1 Comprensió del domini i dels objectius

- **Descripció**: Abans de començar, és essencial entendre el problema que es vol resoldre, el domini d'aplicació i els objectius específics del projecte.
- **Exemple**: En un projecte per predir el consum energètic, cal identificar els factors que influeixen en el consum, com la temperatura, l'ús d'aparells elèctrics, etc.
- **Importància**: Aquesta fase ajuda a definir clarament els objectius i seleccionar les dades més rellevants.

3.4.2 Recollida i pre-processament de dades

- **Descripció**:
    - **Recollida de dades**: Obtenir un conjunt de dades representatiu del problema.
    - **Integració i selecció**: Combinar dades de diferents fonts, eliminant duplicats o informació irrellevant.
    - **Neteja**: Tractar valors nuls, dades sorolloses o fora de rang.
    - **Transformació i normalització**: Escalar les dades perquè tinguin magnituds comparables i codificar variables categòriques si cal.
- **Exemple**: En un projecte de reconeixement facial, s'han de pre-processar les imatges per tenir una resolució uniforme.
- **Importància**: La qualitat del model depèn directament de la qualitat de les dades utilitzades.

3.4.3 Construcció i selecció del model

- **Descripció**:
    - Seleccionar un algorisme d'aprenentatge adequat (p. ex., regressió, arbres de decisió, xarxes neuronals).
    - Ajustar els hiperparàmetres del model per optimitzar el rendiment (p. ex., valors de k en KNN, profunditat màxima en arbres de decisió).
- **Exemple**: Per a un problema de classificació de correu brossa, es pot utilitzar un model Naive Bayes.
- **Importància**: Aquesta fase inclou provar diferents models i optimitzar-los mitjançant tècniques com la validació creuada. Mes detalls a la secció 3.11.

3.4.4 Validació i interpretació dels resultats

- **Descripció**:
    - Dividir les dades en conjunts d'entrenament, validació i test.
    - Mesurar el rendiment amb mètriques adequades (precisió, recall, F1-score, etc.).
    - Identificar possibles problemes com el sobreajustament (overfitting) o el subajustament (underfitting). Mes a la secció 3.12.
- **Exemple**: Utilitzar un conjunt de validació per ajustar els hiperparàmetres i després avaluar el model final amb el conjunt de test.
- **Importància**: Aquesta etapa assegura que el model generalitza bé en dades noves.

3.4.5 Iteració i millora contínua

- **Descripció**:
    - Revisar els passos anteriors per identificar àrees de millora.
    - Incorporar noves dades, millorar el pre-processament o ajustar el model.
- **Exemple**: Si el model de predicció de preus d'habitatges té baix rendiment, es pot afegir informació addicional com la ubicació o la proximitat a serveis.
- **Importància**: L'aprenentatge automàtic és un procés iteratiu que requereix refinament constant.

3.4.6 Desplegament i manteniment

- **Descripció**:
    - Implementar el model en un entorn de producció.
    - Monitoritzar el rendiment amb dades en temps real i actualitzar-lo quan sigui necessari.
- **Exemple**: En un sistema de recomanacions, el model ha d'adaptar-se a les preferències canviants dels usuaris.
- **Importància**: Aquesta fase assegura que el model sigui útil i es mantingui efectiu al llarg del temps.

3.5 Altres mètodes de reducció de dimensionalitat

La reducció de dimensionalitat és una tècnica essencial per simplificar conjunts de dades complexos, permetent la seva visualització i l'aplicació d'algorismes de Machine Learning de manera més eficient. Aquesta tècnica busca transformar un conjunt de dades amb moltes característiques en una representació amb menys dimensions, mantenint el màxim d'informació rellevant.

Objectius principals

- **Visualització**: Representar dades de múltiples dimensions en espais bidimensionals o tridimensionals per detectar patrons.
- **Eficiència computacional**: Reduir la càrrega computacional en algorismes que operen amb conjunts de dades grans.
- **Eliminació de soroll**: Filtrar característiques menys rellevants que poden afegir complexitat o confusió als models.

Mètodes principals

1. **Anàlisi de components principals (PCA)**:
    - **Descripció**: Projecta les dades en un espai de menor dimensió maximitzant la variabilitat explicada.
    - **Aplicació**: Molt utilitzat en visualització i pre-processament de dades amb moltes característiques correlacionades.
    - **Limitacions**: Assumeix que les dimensions rellevants són lineals, cosa que pot ser problemàtica en dades més complexes.
2. **T-SNE (t-Distributed Stochastic Neighbor Embedding)**:
    - **Descripció**: Una tècnica no lineal que transforma dades d'alta dimensionalitat en un espai de dues o tres dimensions tot mantenint la proximitat entre punts similars.
    - **Aplicació**: Visualització de dades complexes com imatges o dades genètiques.
    - **Limitacions**: Elevat cost computacional i dependència de paràmetres com el "perplexity".
3. **UMAP (Uniform Manifold Approximation and Projection)**:
    - **Descripció**: Similar al T-SNE, però més eficient i amb millor preservació de la globalitat de l'estructura de les dades.
    - **Aplicació**: Dades de grans volums on la velocitat i la conservació de l'estructura global són essencials.
    - **Limitacions**: Com T-SNE, no és trivial seleccionar els hiperparàmetres adequats.
4. **Linear Discriminant Analysis (LDA)**:
    - **Descripció**: Projecta dades en un espai de menor dimensionalitat maximitzant la separació entre classes etiquetades.
    - **Aplicació**: Específicament útil en problemes de classificació supervisada.
    - **Limitacions**: Requereix dades etiquetades i assumeix que les dades segueixen una distribució normal.
5. **Autoencoders (model basat en xarxes neuronals)**:

- **Descripció**: Xarxes neuronals entrenades per reconstruir les dades d'entrada a partir d'una representació comprimida (latent space).
- **Aplicació**: Dades no lineals o altament complexes.
- **Limitacions**: Necessita grans conjunts de dades i entrenament computacionalment intensiu.

3.6 Problemes ètics en ML

1. **Privacitat de les dades**
   - Els sistemes d'IA sovint depenen de dades sensibles, com ara informació mèdica o financera. La manipulació inadequada d'aquestes dades pot posar en risc la privacitat dels usuaris.
   - Exemple: El reconeixement facial utilitzat per a vigilància pot recopilar informació personal sense el consentiment explícit.
2. **Biaixos en els models**
   - Si les dades d'entrenament contenen biaixos, el model pot perpetuar o amplificar aquests biaixos, conduint a discriminació.
   - Exemple: Algoritmes de selecció de personal poden discriminar grups subrepresentats si les dades històriques reflecteixen biaixos de gènere o raça.
3. **Transparència i explicabilitat**
   - Molts algorismes, especialment els de xarxes neuronals profundes, funcionen com "caixes negres", dificultant entendre com arriben a les seves decisions.
   - Això pot generar desconfiança en sectors crítics com la medicina o la justícia.
4. **Impacte social**
   - L'automatització pot substituir llocs de treball humans, creant desigualtats econòmiques.
   - Exemple: Sistemes de conducció autònoma que poden eliminar feines de transport.

3.7 Mètriques d'avaluació i selecció de models

Per avaluar el rendiment dels models d'aprenentatge automàtic i seleccionar el model més adequat, s'utilitzen diverses mètriques i tècniques, especialment en contextos de dades desequilibrades o problemes complexos.

3.7.1 Mètriques principals

- **Precisó (Precision):** Proporció de prediccions positives correctes respecte al total de prediccions positives.
  - Fòrmula:
  - Aplicació: Indicada en escenaris com la detecció de fraus.
- **Record (Recall o Sensitivity):** Proporció de casos positius detectats pel model respecte al total de casos positius.
  - Fòrmula:
  - Aplicació: Utilitzada en diagnòstics mèdics.
- **Especificitat (Specificity):** Proporció de negatius correctament classificats respecte al total de casos negatius.

    - Fòrmula:
- **Exactitud (Accuracy):** Proporció de prediccions correctes respecte al total de casos.
    - Fòrmula:
- **F1-Score:** Mitjana harmònica de la precisó i el record.
    - Fòrmula:

3.7.2 Selecció de models

- **Divisió Train-Test:** Divisió de les dades en conjunts d'entrenament (70%) i test (30%). Això ajuda a estimar el rendiment general.
- **Validació creuada (Cross-Validation):** Divisió del conjunt de dades en particions. Es fa rotació entre entrenament i validació per obtenir estimacions més robustes. Valor típic de : 10. Mes detalls a la secció 3.11.
- **Mètriques en dades desequilibrades:** En contextos amb una classe dominant, es prioritzen la precisó i el record sobre l'exactitud.

3.8 Regresió lineal detallada

La regressió lineal és un model matemàtic utilitzat per establir relacions entre variables i fer prediccions. Representa una base fonamental de molts models més complexes.

3.8.1 Definició

Un model de regressió lineal ajusta una línia recta als punts de dades per minimitzar la distància entre els valors predits i els valors reals. La seva forma general:

3.8.2 Construcció del model

- **Funció de cost:**
    - Es basa en la suma dels errors quadràtics (MSE):
    - Aquesta funció permet minimitzar la diferència quadràtica entre les prediccions i els valors reals. Els coeficients s'ajusten per trobar el mínim d'aquesta funció.
- **Optimització:**
    - Els coeficients s'ajusten mitjançant mètodes com el descens del gradient. També es poden utilitzar mètodes algebraics, com la inversa de la matriu de disseny, per problemes simples.
- **Aplicacions:**
    - Models predictius bàsics com l'estimació del preu d'habitatges, on els coeficients s'interpreten com la influència de cada variable (ex. superfície o proximitat al transport públic).

3.8.3 Limitacions

- **Relacions no lineals:**

    - La regressió lineal només captura relacions lineals, fet que la limita en contextos més complexos.
- **Sensibilitat al sobreajustament:**
    - Si el model és massa flexible, pot ajustar-se massa al conjunt d'entrenament, perdent capacitat de generalitzar. Això es resol amb regularització (veure secció 3.7).
- **Multicolinearitat:**
    - Quan les variables explicatives estan altament correlacionades, pot dificultar la interpretació dels coeficients i la robustesa del model.

3.8.4 Variants i millores

- **Regressió polinòmica:** Extensió per modelar relacions no lineals utilitzant termes elevats a potències.
- **Regressió Ridge i Lasso:** Tècniques de regularització per evitar el sobreajustament. Mes a la secció 3.12.

3.9 Xarxes neuronals artificials

Les xarxes neuronals artificials (ANN) estan inspirades en el funcionament de les neurones biològiques. S'utilitzen per modelar relacions no lineals complexes.

3.9.1 Arquitectura bàsica

- **Neuron:** Un node que rep valors d'entrada , aplica pesos i una funció d'activació, produint una sortida :
    - La funció d'activació (sigmoide, ReLU, etc.) introdueix no linealitat al sistema.
- **Capas:** Organitzades en:
    - **Capa d'entrada:** Rep els atributs d'entrada.
    - **Capas amagades:** Realitzen càlculs intermedis.
    - **Capa de sortida:** Proporciona el resultat final.

3.9.2 Entrenament

- **Funció de cost:** Mesura la diferència entre les prediccions del model i els valors reals. Ex.: MSE per problemes de regressió, entropia creuada per classificació.
- **Descens del gradient:** Mètode d'optimització per ajustar els pesos i minimitzar la funció de cost.

3.9.3 Aplicacions

- **Visió per computador:** Reconeixement facial, classificació d'imatges.
- **Processament del llenguatge natural:** Traducció automàtica, anàlisi de sentiment.
- **Jocs i simulacions:** Xarxes com AlphaGo.

3.9.4 Connexió amb models simples

- Les ANN es poden veure com una extensió dels models lineals, afegint complexitat mitjançant capes i funcions d'activació no lineals.
- Les neurones artificials són conceptes bàsics que han evolucionat cap a xarxes profundes (deep learning), augmentant la capacitat de generalització en problemes complexos.

3.10 Hiperparàmetres i regularització

3.10.1 Hiperparàmetres

- **Definició:** Paràmetres definits abans de l'entrenament que controlen el comportament del model.
    - Exemples: Nombre de veïns en KNN (), coeficient de regularització ().
- **Ajustament:** L'elecció d'hiperparàmetres com pot realitzar-se mitjançant cerca exhaustiva (grid search) o optimització bayesiana.

3.10.2 Regularització en regressió

La regularització és una tècnica utilitzada per millorar el rendiment dels models de regressió, especialment en situacions on el model tendeix a sobreajustar-se a les dades d'entrenament. Aquesta tècnica introdueix una penalització al cost associat a la complexitat del model, afavorint solucions més simples que generalitzin millor a noves dades.

Ridge Regression (Regressió Ridge)

La regressió Ridge modifica la funció de cost de la regressió lineal afegint un terme que penalitza la suma dels quadrats dels coeficients del model:

- **Objectiu**: Minimitzar l'error de predicció mantenint els coeficients petits per evitar sobreajustament. Mes a la secció 3.12.
- **Àrea d'aplicació**: Ridge és adequat quan hi ha colinealitat entre els predictors o quan el nombre de predictors supera el nombre d'observacions.
- **Efecte**: Penalitza els coeficients grans, promovent models amb coeficients més uniformes.

Lasso Regression (Regressió Lasso)

La regressió Lasso també afegeix un terme de regularització a la funció de cost, però utilitza la suma dels valors absoluts dels coeficients:

- **Objectiu**: Reduir l'error i simplificar el model eliminant predictors menys rellevants.
- **Característica principal**: La penalització pot portar coeficients a zero, efectivament seleccionant un subconjunt de predictors.
- **Aplicacions**: Ideal per a escenaris on es busca una selecció automàtica de variables.

Diferències clau entre Ridge i Lasso

Característica	**Ridge**	**Lasso**
Penalització	Quadrats dels coeficients	Valors absoluts dels coeficients
Impacte els coeficients	Redueix, però mai porta a zero	Pot portar coeficients a zero
Selecció de variables	No	Sí

Elastic Net

L'Elastic Net combina Ridge i Lasso, afegint una penalització que inclou tant la suma dels quadrats com dels valors absoluts dels coeficients:

- **Objectiu**: Combinar els avantatges de Ridge i Lasso.
- **Aplicacions**: Adequat per dades amb moltes variables correlacionades o on es busca una selecció de variables parcial.

Importància de la normalització de les característiques

En tots els mètodes de regularització, és crucial normalitzar les característiques abans de l'entrenament del model. Sense normalització, les característiques amb escales diferents poden tenir un impacte desproporcionat en la penalització.

Avantatges de la regularització

- Redueix el sobreajustament. Mes a la secció 3.12.
- Millora la capacitat de generalització del model.
- Facilita la interpretació del model amb Lasso, que pot seleccionar un subconjunt de predictors.

Limitacions

- Requereix selecció acurada dels hiperparàmetres (com).
- Pot no ser adequat si totes les variables són igualment rellevants.

La regularització és una eina poderosa per fer front als reptes de sobreajustament, especialment en contextos amb moltes variables predictors o dades sorolloses. L'elecció entre Ridge, Lasso o Elastic Net depèn de les característiques específiques del conjunt de dades i els objectius del modelador.

3.10.3 Beneficis

- Millora la generalització.
- Redueix la complexitat del model.
- Selecciona automàticament les variables més rellevants (Lasso).

3.10.4 Normalització

- **Escalat de dades:**
    - Per evitar que variables amb escales molt diferents dominin la funció de cost, és recomanable normalitzar les dades. Ex.: escalar Min-Max o estandardització (Z-score).
- **Impacte:** Garanteix que tots els atributs contribueixin de manera similar al model, evitant biaixos deguts a magnituds dispars.

3.11 Validació creuada

La validació creuada ("k-fold cross-validation") és una tècnica fonamental en l'avaluació de models d'aprenentatge automàtic, que s'utilitza per assegurar la generalització dels models i evitar problemes com el sobreajustament (*overfitting*).

Objectiu

L'objectiu principal de la validació creuada és utilitzar de manera eficient les dades disponibles per:

- Estimar el rendiment real del model sobre dades desconegudes.
- Optimitzar la selecció de hiperparàmetres.
- Detectar problemes de sobreajustament o subajustament. Mes a la secció 3.12.

Funcionament

1. **Divisió de les dades:**
    - Es divideixen les dades en *k* particions (o plecs) aproximadament iguals.
    - Cada plec serveix successivament com a conjunt de validació, mentre que la resta dels plecs s'utilitzen com a conjunt d'entrenament.
2. **Iteracions:**
    - Es realitzen *k* iteracions. En cada iteració:
        - Es fa servir un plec diferent com a conjunt de validació.
        - Els altres *(k-1)* plecs s'utilitzen per entrenar el model.
3. **Càlcul de l'error:**
    - Es calcula l'error de validació per a cadascun dels *k* plecs.

    - Es pren la mitjana dels errors com a estimació global del rendiment del model.

Variants de la validació creuada

- **Leave-One-Out Cross-Validation (LOOCV):** Cada instància del conjunt de dades serveix com a conjunt de validació (equivalent a $k = n$, on n és el nombre total d'instàncies).
    - Avantatge: Utilitza al màxim les dades disponibles per entrenar.
    - Inconvenient: Pot ser computacionalment costosa.
- **Stratified k-Fold Cross-Validation:** Els plecs es creen mantenint les proporcions de les classes, especialment útil en conjunts de dades desequilibrats.

Beneficis

- **Aprofitament de les dades:** Permet que totes les instàncies siguin utilitzades tant per entrenar com per validar el model.
- **Estimació robusta de l'error:** Redueix la variabilitat dels resultats associada a una única partició d'entrenament/validació.
- **Millor selecció d'hiperparàmetres:** Ajuda a optimitzar valors com el nombre de veïns en KNN o el coeficient de regularització en regressió.

Limitacions

- **Cost computacional:** Requereix entrenar el model k vegades, augmentant significativament el temps de càlcul.
- **Dependència de *k*:** La selecció inadequada del nombre de plecs pot afectar els resultats:
    - Un k petit pot portar a estimacions menys fiables.
    - Un k gran, com en LOOCV, pot ser computacionalment ineficient.

Exemple pràctic

Per exemple, en un model KNN, la validació creuada pot ajudar a determinar el millor valor de k (nombre de veïns) provant diferents valors i seleccionant aquell que minimitza l'error de validació. Això assegura que el model generalitzi millor en dades no vistes.

Conclusió

La validació creuada és una tècnica clau en el desenvolupament de models robustos d'aprenentatge automàtic, que proporciona una estimació fiable del rendiment i facilita la presa de decisions en la configuració del model.

3.12 Sobreajustament i Subajustament

Definicions

1. **Sobreajustament (Overfitting):**
    - Es produeix quan un model aprèn en excés els detalls i el soroll de les dades d'entrenament, perdent capacitat de generalització a dades noves.

    - Exemple: Un model amb molts paràmetres que classifica correctament cada punt de l'entrenament, però falla en fer prediccions fiables sobre dades no vistes.
2. **Subajustament (Underfitting):**
    - Passa quan un model és massa simple i no captura adequadament les relacions entre les dades d'entrenament.
    - Exemple: Una recta simple per ajustar dades que segueixen una relació clarament no lineal.

Símptomes i Causes

- **Sobreajustament:**
    - L'error de l'entrenament és molt baix, però l'error de validació o test és alt.
    - Apareix quan:
        - El model té massa paràmetres en relació amb el volum de dades.
        - Les dades tenen soroll que el model intenta ajustar com si fossin patrons reals.
        - No es fa regularització durant l'entrenament.
- **Subajustament:**
    - Els errors en entrenament i validació són alts.
    - Apareix quan:
        - El model és massa senzill per capturar patrons complexos (p. ex., regressió lineal per dades amb relacions no lineals).
        - Les dades d'entrenament no són suficients o no són representatives.

Solucions

- **Evitar el Sobreajustament:**
    1. **Regularització:**
        - Afegir termes de penalització al cost del model (p. ex., Ridge, Lasso).
    2. **Validació Creuada:**
        - Utilitzar tècniques com el *k-fold cross-validation* per optimitzar el rendiment del model.
    3. **Reduir la complexitat del model:**
        - Disminuir el nombre de paràmetres o capes si és una xarxa neuronal.
    4. **Augmentar dades d'entrenament:**
        - Recol·lectar més dades per oferir al model una millor representació del domini.
- **Evitar el Subajustament:**
    1. **Augmentar la complexitat del model:**
        - Afegir paràmetres o utilitzar models més complexos (p. ex., models no lineals).
    2. **Millorar el preprocessament de dades:**
        - Detectar i solucionar problemes de qualitat, com dades mancants o atributs irrellevants.

3. **Incrementar iteracions d'entrenament:**
    - Permetre que el model aprengui durant més temps.

Indicadors i Diagnòstic

- **Corbes d'aprenentatge:**
    - Representar l'error en funció del nombre d'instàncies o iteracions.
    - En casos de sobreajustament:
        - L'error d'entrenament continua disminuint però l'error de validació comença a augmentar.
    - En casos de subajustament:
        - Els errors d'entrenament i validació es mantenen alts.
- **Prova amb dades noves:**
    - Si el model funciona molt bé amb les dades d'entrenament però falla en dades no vistes, és probable que pateixi sobreajustament.

Importància

Assegurar-se que un model ni sobreajusta ni subajusta és crucial per aconseguir una bona generalització, la qual cosa permet que funcioni adequadament amb dades reals o noves. Això és especialment important en aplicacions crítiques com la detecció de fraus, diagnòstics mèdics o sistemes de predicció financera.

4. Laboratori

4.1 Scatter matrix, plots i histogrames

Importar dades i ensenyar valors:

```
Unset
housing = pd.read_csv("data/housing.csv")
housing.shape
housing.info()
housing.describe()
```

Crear histograma, scatter matrix i scatterplot:

- bins: nombre de trossos en què es dividiran les dades a l'histograma.
- figsize: canviar l'alçada de la figura (com un resize).
- kind: tipus de plot, aquí li diem que sigui de tipus scatter per fer un scatterplot,
- x: valors en el gràfic de les x.
- y: valors en el gràfic de les y.

```
Unset
housing.hist(bins=50, figsize=(20,15))
plt.show()

attributes = ["median_house_value", "median_income", "total_rooms",
"housing_median_age"]
scatter_matrix(housing[attributes], figsize=(12,8))

housing.plot(kind="scatter", x="median_income", y="median_house_value",
alpha=0.1)
```

Afegir nous atributs:

```
Unset
housing["population_per_household"] = housing["population"] /
housing["households"]
housing[housing['population_per_household']>10].shape
scatter_matrix(housing[attributes], figsize=(12,8))
```

4.2 Dividir dades d'entrenament i test i fer servir KNN

Llegir dades i fer plots:

```
Unset
iris = pd.read_csv('data/iris.csv')
setosa = iris[iris.Species=='Iris-setosa']
versicolor = iris[iris.Species=='Iris-versicolor']
virginica = iris[iris.Species=='Iris-virginica']

setosa = iris[iris.Species=='Iris-setosa']
versicolor = iris[iris.Species=='Iris-versicolor']
virginica = iris[iris.Species=='Iris-virginica']

setosa = iris[iris.Species=='Iris-setosa']
versicolor = iris[iris.Species=='Iris-versicolor']
virginica = iris[iris.Species=='Iris-virginica']
```

Dividir dades i entrenar model:

- random_state: llavor que decideix com particionar les dades en training i en test.
- n_neighbors: nombre de veïns en què dividir les dades.

```
Unset
attributes = ['SepalLength', 'SepalWidth', 'PetalLength', 'PetalWidth']
X = iris[attributes]
y = iris['Species']
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(X_train, y_train)
```

Aplicar K-Nearest Neighbor per intentar predir:

```
Unset
df = pd.DataFrame(data=X_test, columns=attributes)
df['true class'] = y_test
df['kNN_pred'] = knn.predict(X_test)
df
```

Obtenir score d'encerts:

```
Unset
knn.score(X_test, y_test)
```

Podem fer un loop que calculi quin és el millor nombre de veïns:

```
Unset
k_range = [num for num in range(110) if num % 10 == 1]
```

```
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors = k)
    knn.fit(X_train, y_train)
    print("with {}-NN accuracy is {}".format(k, knn.score(X_test, y_test)))
```

4.3 Regressió de 3 tipus: quadrats, ridge i lasso

Importem dades i tornem a entrenar:

```
Unset
murders = pd.read_csv('data/murders.txt', sep=" ")
attributes = ['inhabitants','income','unemployment']
X = murders[attributes]
y = murders['murders']

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=55)
```

Apliquem minmaxscaling per normalitzar els valors (per no fer alguns factors de l'error més importants que d'altres):

```
Unset
scaler = MinMaxScaler()
scaler.fit(X_train)
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)

pd.DataFrame(data = X_train_scaled, columns = list(X)).head()
pd.DataFrame(data = X_test_scaled, columns = list(X)).head()
```

Els valors del test_scaled podem arribar a ser majors de 1 inclús després d'haver-los normalitzat abans.
Ara apliquem els 3 tipus de regressió:

```
Unset
alpha = 1

from sklearn.linear_model import LinearRegression, Ridge, Lasso
ols = LinearRegression().fit(X_train, y_train)
lasso = Lasso(alpha = alpha).fit(X_train, y_train)
ridge = Ridge(alpha = alpha).fit(X_train, y_train)

ols_scaled = LinearRegression().fit(X_train_scaled, y_train)
```

```
lasso_scaled = Lasso(alpha = alpha).fit(X_train_scaled, y_train)
ridge_scaled = Ridge(alpha = alpha).fit(X_train_scaled, y_train)
```

4.4 Regressió lineal i cross-validate

Importem dades i fem el training com abans:

```
Unset
columns = "age sex bmi map tc ldl hdl tch ltg glu".split()
diabetes = datasets.load_diabetes() # Call the diabetes dataset from sklearn
df = pd.DataFrame(diabetes.data, columns=columns) # load the dataset as a
pandas data frame
y = diabetes.target # define the target variable (dependent variable) as y

X_train, X_test, y_train, y_test = train_test_split(df, y, test_size=0.2)
```

Preveure resultats fent servir un model lineal:

```
Unset
lm = linear_model.LinearRegression()
model = lm.fit(X_train, y_train)
predictions = lm.predict(X_test)
plt.scatter(y_test, predictions)
plt.xlabel("True Values")
plt.ylabel("Predictions")
plt.title("Model score  = {}".format(model.score(X_test, y_test)))
plt.show()
```

Veiem que el scatterplot no té una línia recta massa definida. Ara fem el mateix entrenant fent servir cross-validation, i veurem que el scatterplot té una línia recta molt més clara i definida amb més punts.

```
Unset
scores = cross_val_score(model, df, y, cv=10)

kf = KFold(n_splits=10, shuffle=True)
predictions = cross_val_predict(model, df, y, cv=kf)
plt.scatter(y, predictions)
plt.xlabel("True Values")
plt.ylabel("Predictions")
plt.title("Model score  = {}".format(metrics.r2_score(y, predictions)))
plt.show()
```

4.5 Tunning d'hiperparàmetres

Importem dades de diabetes i tenim diferents valors de alfa a provar:

```
Unset
dataset = datasets.load_diabetes()
alphas = np.array([1,0.1,0.01,0.001,0.0001,0])

# split into train and test as usual..
X_train, X_test, y_train, y_test = train_test_split(dataset.data,
dataset.target)
```

Provem cada valor de alfa fent servir el mètode de ridge:

```
Unset
grid = GridSearchCV(estimator=Ridge(), param_grid=dict(alpha=alphas), cv=10)
grid.fit(X_train, y_train)
```

La millor alfa es troba fent servir best_estimator_:

```
Unset
model = Ridge(alpha=grid.best_estimator_.alpha).fit(X_train, y_train)
```

4.6 Trobar nombre de clusters òptim

```
Unset
nc1, nc2 = 3, 4
kmeans1 = KMeans(n_clusters = nc1)
kmeans1.fit(iris.data)
score1 = silhouette_score(iris.data, kmeans1.labels_)
kmeans2 = KMeans(n_clusters = nc2)
kmeans2.fit(iris.data)
score2 = silhouette_score(iris.data, kmeans2.labels_)
```