Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Enginyeria Informàtica

Títol: Eines d'Intel·ligència Artificial per a la predicció del COVID-19

Document: Memòria

Alumne: Varsha Tolani

Tutor: Xavier Lladó Bardera i Robert Martí Marly
Departament: Arquitectura i Tecnologia de Computadors
Àrea: Arquitectura i Tecnologia de Computadors

Convocatòria (mes/any): Juny 2021

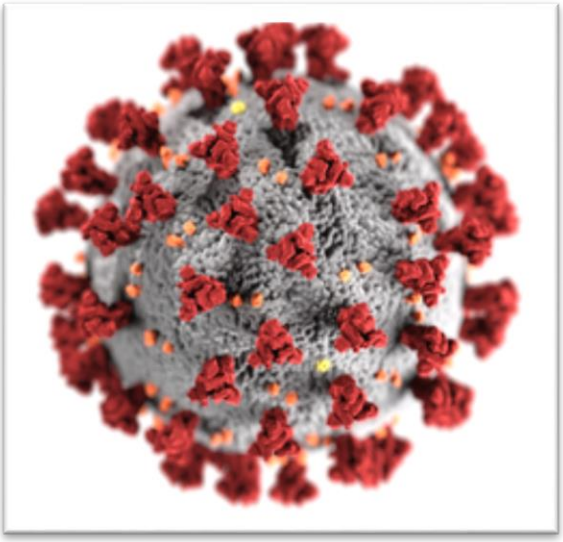
Índex

1 Introducció, planificació i conceptes

1.1 Introducció

Aquest últim any s'ha viscut una situació que ha agafat a la població mundial per sorpresa, ningú estava preparat per una expansió tan ràpida de la malaltia de la **COVID-19**.

Es va detectar per primer cop el desembre de 2019 a la Xina, es tracta d'una malaltia infecciosa també coneguda com a SARS-CoV-2 (veure Il·lustració 1), que provoca complicacions similars a una **pneumònia** i causa **dificultats respiratòries**.



Il·lustració 1. Virió del virus SARS-CoV-2
Wikipedia, "SARS-CoV-2", 2020

Els símptomes més habituals inclouen: febre, tos, fatiga, pèrdua de l'olfacte i el gust, mal de cap; molt semblants als de la grip. La majoria de les persones contagiades passen la malaltia de manera lleu, i alguns sobretot, son asimptomàtics (concentrats en la població jove), però en els pitjors dels casos, els pacients requereixen ser ingressats a les unitats de vigilància intensiva i tenen un alt risc de defunció.

Fins a data d'avui, aquesta malaltia ha afectat unes **170.000.000 persones** i ha causat més de **3.000.000 morts**, que ha suposat un gran impacte pel mon i per a la comunitat mèdica, per la qual el sector informàtic ha vist l'oportunitat d'avançar i créixer per donar-li tot el suport possible.

Les aplicacions informàtiques en la medicina es van començar a utilitzar amb molta anterioritat, i han demostrat la seva capacitat d'ajudar als metges a gestionar la informació dels seus pacients, connectar-se de manera més eficient amb altres departaments interns, i sobretot, a diagnosticar i prevenir malalties.

1.1.1 Motivacions

El desenvolupament d'aquest projecte neix de la necessitat d'ajudar durant el procés de diagnosi i va enfocat a crear una nova aplicació basada en la Intel·ligència Artificial, per donar suport als hospitals en l'actual pandèmia.

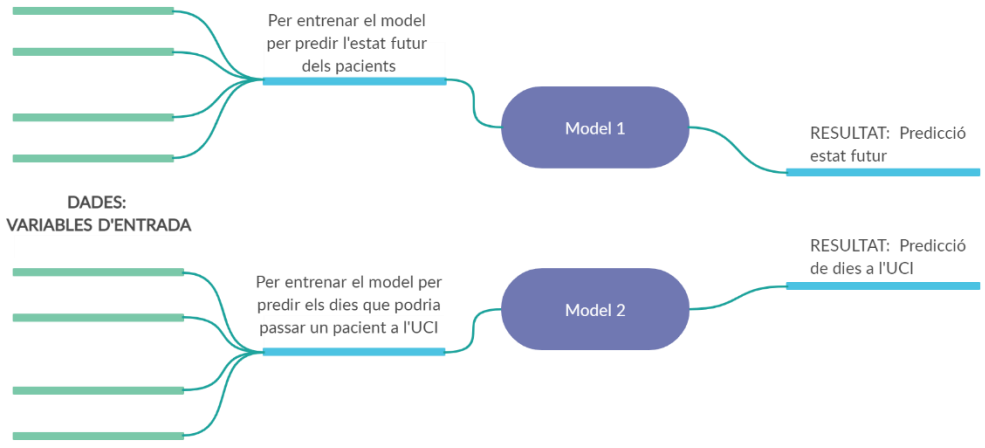
Específicament, permetrà al personal mèdic generar prediccions sobre les possibles evolucions dels pacients durant les etapes de la malaltia. Això els ajudarà a prendre decisions respecte el tractament que ha de rebre cadascun dels pacients o saber si en un futur requerirà ingressar a la UCI, el qual, com hem vist, és un punt molt important a conèixer per poder gestionar la pressió sanitària als hospitals.

1.1.2 Propòsit i objectius del projecte

L'objectiu principal del model final és clarament alleujar una mica la pressió sanitària, proporcionant tota l'ajuda possible a l'hora de fer diagnosis sobre la malaltia i preveure l'estat futur d'un pacient.

Per això, el producte final, constarà de dos algorismes (veure Il·lustració 2), el principal predirà l'estat de sortida d'un pacient contagiat, que actualment indica si el pacient tornarà a casa, seguirà ingressat o si tindrà risc de defunció.

Un segon model proporcionarà una altra funcionalitat, fer un pronòstic sobre els dies que un pacient podria passar a l'UCI, en cas que ja hagi sigut ingressat. D'aquesta manera, es vol donar temps als metges per tenir el control sobre l'ocupació i disponibilitat de llits.



Il·lustració 2. Esquema dels dos models que formen el projecte

Pel funcionament del model, es disposarà d'una base de dades mèdica, proporcionada pel grup hospitalari ***HM Hospitales***, que consta amb imatges de radiografies i diferents taules que inclouen: constants vitals, medicaments subministrats, data i hora dels subministraments, entre altres, d'individus contagiats i ingressats.

Aquesta informació servirà de base per a entrenar al programa i familiaritzar-lo amb les característiques de la malaltia. Per fer-ho es farà un estudi previ dels diferents algorismes existents de *Machine Learning* (aprenentatge automàtic), una branca de la Intel·ligència Artificial, que consisteix en l'anàlisi de variables per localitzar mostres i trobar comportaments comuns per poder classificar particularitats en les dades.

Un cop entrenat, es procedirà a l'etapa de proves, per assegurar el seu funcionament correcte i el desplegament a un programa real.

Aquest treball es porta a terme conjuntament amb dos professors que es dediquen a l'anàlisi de dades i imatges mèdiques per obtenir pronòstics de diferents malalties o condicions, emmarcat dins d'un projecte desenvolupat pel grup de recerca en Visió per Computador i Robòtica (VICOROB) de la Universitat de Girona i en col·laboració de l'Hospital Dr. Josep Trueta de Girona.

La seva participació donarà una percepció d'aplicació al mon real, on es tracta amb la vida dels pacients. A més a més, es contactarà amb diferents hospitals per conèixer les necessitats específiques del punt de vista dels professionals sanitaris.

Abans d'avançar, s'han de mencionar una sèrie de petits objectius que s'han de complir durant el procés de desenvolupament del projecte:

1. Descarregar i analitzar la base de dades per a **normalitzar** el seu contingut.
2. Familiaritzar-se amb **mètodes** de la Intel·ligència Artificial.
3. Crear els **dos models** de predicció: per classificar el futur estats dels pacients i fer una predicció dels dies que podrien passar a la UCI.
4. **Validar i comparar** els resultats obtinguts de proves fetes amb models experimentals.
5. Elaboració d'aquesta **memòria**.

I finalment, es volen compartir els models creats pel seu ús, per això, s 'ha decidit pujar tant el codi com els models a una plataforma pública: ***GitHub***.

1.2 Base de dades

Els models de IA requereixen d'una base de dades per portar a terme el procés d'entrenament, per aquest projecte s'ha utilitzat un *Data Set* proporcionat per ***HM Hospitales*** (veure Il·lustració 3).

Il·lustració 3. Icona HMH HM Hospitales, 2012

És un grup universitari fundat pel **Dr. Juan Abarca Camal** el 1989, amb l'objectiu de començar un projecte per crear un nou model de medicina privada a Espanya.

Treballen en àrees com l'Oncologia, la Cardiologia, les Neurociències i la Fertilitat, i compten amb 42 centres repartits pel país.

Les variables clíniques que posen a la disposició son totalment anònimes i estan obertes a ser utilitzades per institucions sanitàries, universitats o entitats científiques amb l'objectiu d'avançar en el coneixement del virus.

Les dades que en formen part inclouen: diagnòstics, tractaments, ingressos, estades a la UCI, constants d'urgències, entre d'altres. I tota aquesta informació està repartida en sis taules:

Taula	**Contingut**
Taula principal	Dades del pacient, dades d'urgències, estada la UCI i registres del primer i últim conjunt de constants preses.
Taula de medicació	Informació sobre els medicaments subministrats als pacients, especificant la seva data i el codi d'identificació.
Taula de constants vitals	Registre complet de constants vitals preses durant la estada dels pacients a l'hospital.
Taula de laboratori	Resultats de laboratori de peticions realitzades de cada pacient durant el seu ingrés.
Taula de codificacions	Registres de diagnòstics i procediments codificats segons la classificació internacional CIE10.

Aquestes dades permeten al model conèixer l'estat dels pacients en un moment donat i quines variables estan associades amb ell. Per aquest projecte s'ha fet servir la **Taula Principal**, que conté al voltant de 2000 registres i en els propers apartats s'explica tot el procediment de normalització i el seu ús.

A més d'aquestes taules, la base de dades conté també un **conjunt d'imatges** de radiografies preses de pacients de la malaltia. Aquestes podrien servir per a crear un model de detecció basat en anàlisi d'imatge, però en aquest projecte es treballarà només amb les taules de les variables clíniques.

1.3 Planificació

Per a la planificació del projecte, s'ha dividit el treball en tasques, que ha ajudat a generar blocs segons el tipus de feina a realitzar, permetent seguir perfectament la metodologia en cascada.

1.3.1 Paquets de treball

A continuació es mostra l'esquema dels paquets de treball (veure Il·lustració 4), que alhora es divideixen en sub-tasques. L'ordre que es mostra a la imatge és d'esquerra a dreta, és a dir, que per començar un paquet prèviament s'ha d'haver completat el de l'esquerra.

Il·lustració 4. Esquema de paquets de treball

1.3.2 Descripció de paquets

En aquest apartat es detalla què es vol obtenir de cada paquet, ja que com s'ha mencionat anteriorment, cada etapa espera unes entrades de la seva antecessora.

Per tal de portar un control del temps, s'ha fet una estimació en setmanes de la duració de cadascun d'ells:

Nom del paquet	Documentació [DOC]
Descripció	Redacció dels detalls i procés seguit i tasques portades a terme per crear el programa.
Temporalització	Durant tot el temps del projecte.
Tasques	Redacció de la memòria.
Resultats	Document de la memòria.

Nom del paquet	Anàlisi de requisits [REQ]
Descripció	Anàlisi de qui i com usarà el programa per desenvolupar el model adaptat a les necessitats.
Temporalització	2 setmanes.
Tasques	Anàlisi d'usuaris i utilitats.
Resultats	Requisits definits.

Nom del paquet	Entrenament i pràctica [EiP]
Descripció	És un període d'aprenentatge per familiaritzar-se amb llibreries i algorismes per *machine learning*, i també practicar amb mètodes d'anàlisi d'imatge.
Temporalització	4 setmanes.
Tasques	Estudi d'algorismes de *machine learning* i coneixement de mètodes d'anàlisi d'imatge.
Resultats	Coneixement de l'àmbit suficient per començar el projecte.

Nom del paquet	Dades [DADES]
Descripció	Recopilació de dades per a entrenar el model.
Temporalització	2 setmanes.
Tasques	Agrupació de dades, neteja i normalització.
Resultats	Dades preparades per entrenar el model.

Nom del paquet	Model [MOD]
Descripció	Creació i desenvolupament de l'eina de predicció.
Temporalització	4 setmanes.
Tasques	Crear el model, analitzar diferents algorismes i entrenament.
Resultats	Model preparat i llest pel *testing*.

Nom del paquet	Testing [TEST]
Descripció	Realització de proves de funcionament al model.
Temporalització	4 setmanes.
Tasques	Comprovació de resultats obtinguts amb els reals.
Resultats	Model amb funcionament correcte i verificat.
Nom del paquet	Disseny [DIS]

Descripció	Dissenyar l'entorn web on els usuaris accediran per generar prediccions.
Temporalització	1 setmana.
Tasques	Disseny del FrontEnd.
Resultats	Pàgina web dissenyada.

Nom del paquet	Desplegament [DESP]
Descripció	Pujar el programa a la web pel seu ús.
Temporalització	1 setmana.
Tasques	Desplegar el projecte a internet i escriure la seva documentació.
Resultats	Model desplegat.

1.3.3 Matriu de traçabilitat

Per portar un registre de dependències s'ha creat una matriu de traçabilitat, que ajuda a visualitzar quins requisits s'han de complir per poder satisfer d'altres. En el cas de la documentació, com és un procés que es porta a terme durant tot el projecte, depèn de tots ells:

	Doc.	Requisits	Entrenament	Dades	Model	Test	Disseny	Desplegar
RFD1	√		√	√				
RFD2	√		√	√				
RFD3	√		√	√	√			
RFD4	√		√	√	√			
RFD5	√		√		√			
RFD6	√	√	√	√	√			
RFD7	√	√	√	√	√		√	
RFD8	√	√	√				√	√
RFT1	√		√	√		√		
RFT2	√	√	√	√	√	√		
RFT3	√		√		√	√		
RFE1	√		√	√	√			
RFE2	√	√	√	√	√			
RFE3	√	√	√	√	√			
RFE4	√	√	√	√	√		√	
RFC1	√	√	√	√				
RFC2	√	√	√				√	√
RFC3	√	√	√				√	√
RFC4	√	√	√				√	√

1.3.4 Diagrama de Gantt

El diagrama de Gantt és un mètode que permet representar el temps previst per a les tasques d'un projecte. S'ha utilitzat per mostrar setmanalment quin paquet es vol treballar:

2021	*Març*				*Abril*				*Maig*				*Juny*	
Setmana	1	2	3	4	5	6	7	8	9	10	11	12	13	14
[DOC]	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria	Memòria
[REQ]	Identificar usuaris i usos	Definir requisits												
[EiP]			Estudi de llibreries	Pràctica amb algorismes	Pràctica amb algorismes	Conèixer mètodes								
[DADES]							Recol·lectar i normalitzar dades	Organitzar dades	Verificar dades					
[MOD]										Provar diferents mètodes	Triar mètodes	Codificar	Entrenar	
[TEST]										Separar dades	Verificar resultats	Detectar errors	Verificar resultats	
[DIS]														Disseny Front-End
[DESP]														Desplegar

2 Marc de treball i conceptes previs

Aquest projecte abasta temes complexos i d'investigació, amb els quals no tothom podria estar familiaritzat, com son la intel·ligència artificial i la importància de les variables que ajuden a detectar la COVID-19.

Per aquesta raó, en aquest apartat s'expliquen en detall els punts més importants per a que qualsevol persona pugui seguir i entendre'l sense problemes.

2.1 Què és la Intel·ligència Artificial?

La **Intel·ligència Artificial** (IA) és un concepte que avui en dia veiem a tot arreu, des dels assistents de telèfons mòbils fins a models complexos que s'utilitzen en el camp de la recerca. Però, és realment una idea del món modern?

La IA va néixer arran de la pregunta: "*És possible que una màquina actuï de forma intel·ligent*?", proposada pel considerat pare de la informàtica, **Alan Turing** (1912-1954). Tot i així, no va ser fins el 1956 que es va definir el terme per **John McCarthy** (1927-2011) al Darmouth College, que el va voler diferenciar del camp de la cibernètica.

Hi ha moltes i diverses definicions, però la més adequada seria la del seu propi creador:

> "*La Intel·ligència Artificial és la ciència i enginyeria de crear màquines intel·ligents [...], on la intel·ligència és la part computacional de l'habilitat de complir objectius en el món.*" – John McCarthy

Per tant, el que es busca, i el que totes les definicions tenen en comú, és que una màquina sigui capaç d'**imitar el comportament humà**.

Per què és important la IA?

Ara que s'ha explicat què és, es podria preguntar: "per què és important la IA?" o "on i com s'utilitza?".

Doncs bé, les respostes serien molt similars, ja que és capaç de realitzar processos humans, però de manera més **ràpida** i **eficient**, amb l'habilitat analitzar gran quantitat d'informació en poc temps.

Ens serveix perquè:

- Automatitza accions repetitives d'aprenentatge mitjançant dades.
- Pot analitzar més dades i més complexes que els humans.
- Té un comportament adaptatiu a mesura que va aprenent, degut a algorismes d'aprenentatge progressius.
- És capaç de treure una gran exactitud de predicció a través de dades analitzades.
- Extreu el màxim d'informació de les dades de les quals disposa.

I per això, avui en dia, s'utilitza en una gran varietat de camps, alguns dels quals serien:

- **Assistència virtual:** Per automatitzar l'atenció i ajuda al client.
- **Automobilística:** Per proporcionar vehicles robotitzats, capaços de moure's sense conductors.
- **Ciberseguretat:** Per reconèixer i actuar contra atacs informàtics.
- **Salut:** Per ajudes en diagnosis i estudi de tendències mèdiques.
- **Agricultura:** Per augmentar la productivitat al camp i treballar en condicions de risc.

Branques de la IA

La IA té diverses aplicacions i disposa de vàries tècniques com ho podrien ser la lingüística, la visió, planificació, robotització, entre altres. Per tal d'aprofundir en cadascuna d'elles i es separa en branques especialitzades:

***Machine Learning* (Aprenentatge Automàtic)**

L'aprenentatge automàtic és una tècnica que dona les màquines la capacitat d'aprendre sense haver sigut programades, sinó permetent fer-ho per si mateixes.

Aquesta branca permet fer classificacions, estimacions numèriques o desxifrar dades a partir d'un conjunt de dades. I al mateix temps, es classifica en aprenentage **supervisat, no supervisat** i ***deep learning***.

***Natural Language Processing* (Processament de Llenguatge Natural)**

Aquesta branca estudia les interaccions entre les màquines i la parla humana, ja que es capaç d'interpretar el llenguatge parlat. Es porten a terme funcions com ara extreure, classificar, traduir, respondre o generar text, fent possible que les persones puguin comunicar-se mitjançant les màquines.

***Planning and Scheduling* (Planificació)**

També referida com a Planificació Automàtica, aquesta disciplina s'utilitza quan un sistema busca l'assoliment d'objectius, ja que s'encarrega principalment de la producció de plans, dirigits, per exemple, a robots, per la planificació de vehicles de transport o a l'organització del personal d'empresa.

***Expert Systems* (Sistemes Experts)**

Els sistemes experts fan referència als models que actuen com els humans a l'hora de prendre decisions, basant-se en el coneixement del que disposen mitjançant mètodes de raonament.

Es fan servir per solucionar problemes complexos i destaquen per ser molt responsius, confiables i amb una alta capacitat d'execució.

***Speech* (Parla)**

Els sistemes de parla, com indica el seu nom, treballen amb el llenguatge parlat, traduint-los a text i viceversa. D'aquesta manera, permet per exemple, reconèixer usuaris a través de la seva veu, o transformar la veu en diferents idiomes a text per ser interpretada per màquines.

***Robotics* (Robòtica)**

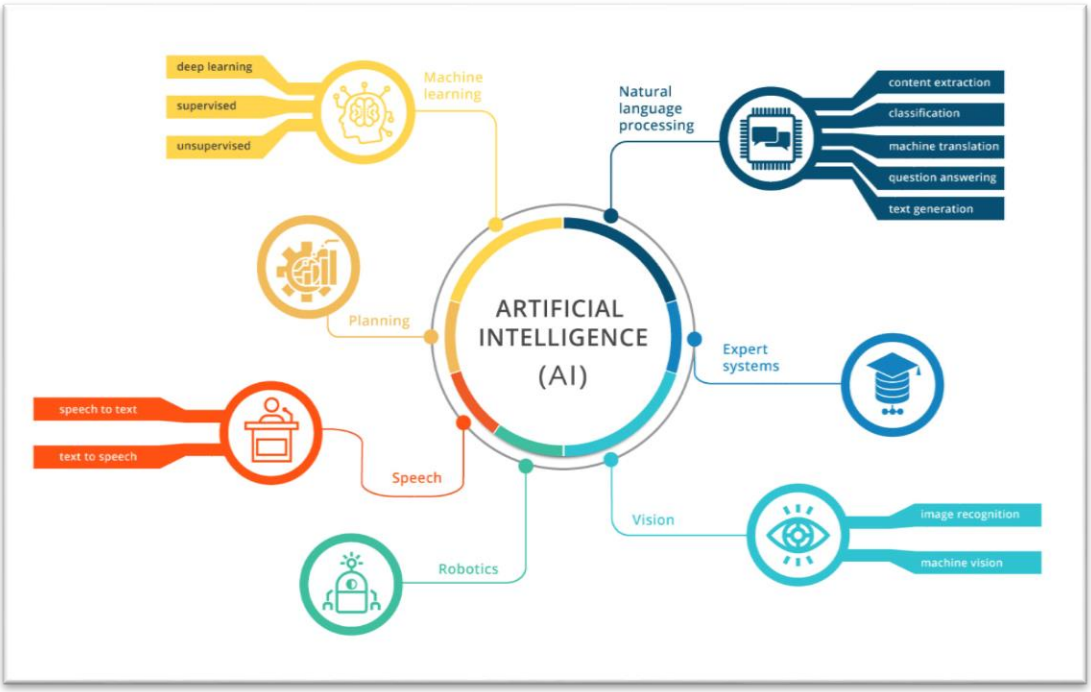
La robòtica té com a objectiu principal el disseny i la construcció de robots.

Busca que aquests puguin portar a terme tasques humanes per automatitzar processos complicats. Un exemple en seria l'ús dels robots en la manufactura d'automòbils.

***Vision* (Visió)**

Els sistemes de visió tracten principalment amb imatges, utilitzant mètodes de processament i d'anàlisi amb l'objectiu d'interpretar-les, anomenant al procés **reconeixement d'imatge.** Es basen en el desglossament per a poder categoritzar les parts que formen una representació i prendre decisions sobre elles.

A continuació, es mostra en una imatge un resum (veure Il·lustració 5) de les diferents branques explicades:



Il·lustració 5. Branques de la Intel·ligència Artificial
Technology Evaluation Centers, "How AI is Transforming ERP", 2021

Per aquest projecte, *Machine Learning* és la branca de la IA que s'ha triat per a crear el model, i per tant, s'explicarà en detall en l'apartat següent (veure 2.2.1 *Machine Learning*).

2.1.1 *Machine Learning*

El *Machine Learning*, o aprenentatge automàtic, és una de les branques de la Intel·ligència Artificial, com s'ha explicat a l'apartat anterior.

Va aparèixer també durant els anys 50, específicament l'any **1959**, quan Arthur Samuel (1901-1990) li va donar el nom i el va definir com:

> "*Un camp d'estudi que li dona a les màquines la capacitat d'aprendre sense haver sigut explícitament programats*" - Arthur Samuel

La paraula clau és **aprendre**, que fa referència a estudiar grans conjunts de dades i fer prediccions a través de la experiència, amb la qual millora un model. Per tant, es

podrien considerar com a mètodes que identifiquen patrons en la informació de la qual disposen i la **categoritzen** o classifiquen per **predir comportaments**.

Aquestes prediccions son les que fan a l'aprenentatge automàtic un camp important, ja que son fetes extraient variables valuoses d'una gran quantitat de dades.

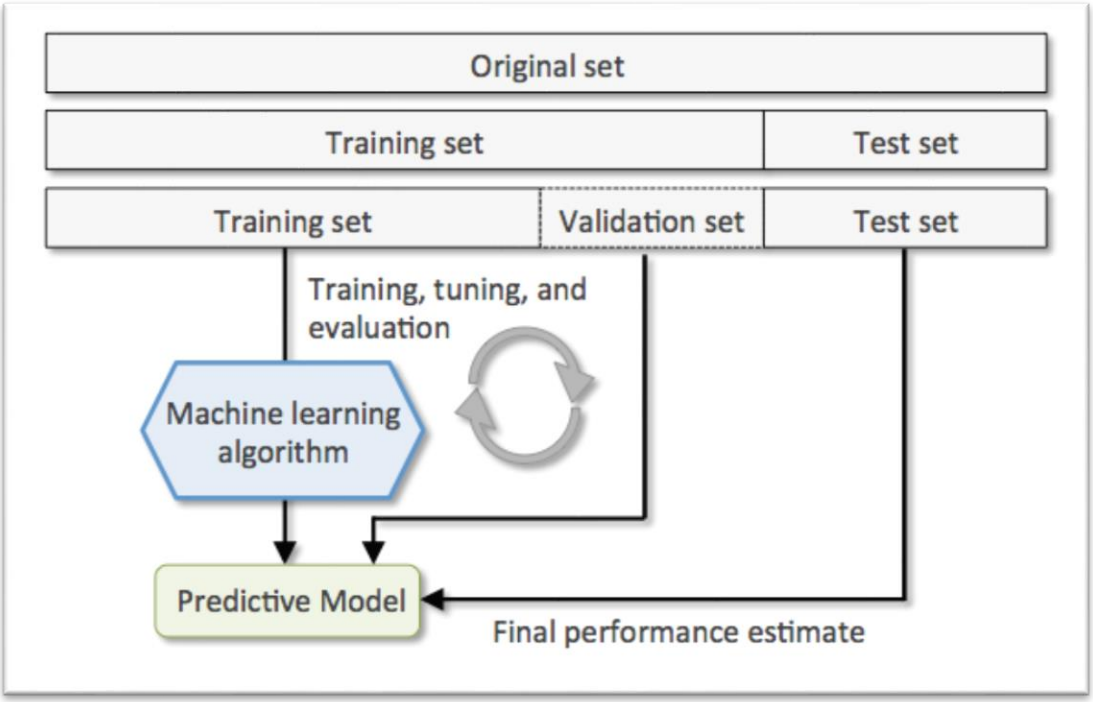
A l'igual que la IA, també es troben diferents enfocs dins d'aquesta branca, que permeten solucionar problemes de diferents tipus:

2.1.1.1 Aprenentatge Supervisat

L'aprenentatge supervisat tracta d'un model matemàtic que és **observat** durant el procés d'aprenentatge.

Consisteix a proporcionar-li un conjunt de dades que contenen tant les variables d'entrada com les de sortida, és a dir, una classificació o predicció esperada. Quan aquestes sortides son etiquetes o classes, es diu que es tracta de **classificació**, i al contrari, quan son valors numèrics, es parla de **regressió**.

Normalment, a aquestes dades se les anomena ***Train Set***, que permeten al model entrenar-se coneixent sempre el resultat, i un cop fet, es comprova el seu funcionament sobre el ***Test Set***, que conté informació no coneguda i sobre el qual es fa la predicció (veure Il·lustració 6). A la imatge següent mostra el repartiment:



Il·lustració 6. Entrenament, validació i testeig
DataLab, "Cross Validation & Ensembling", 2020

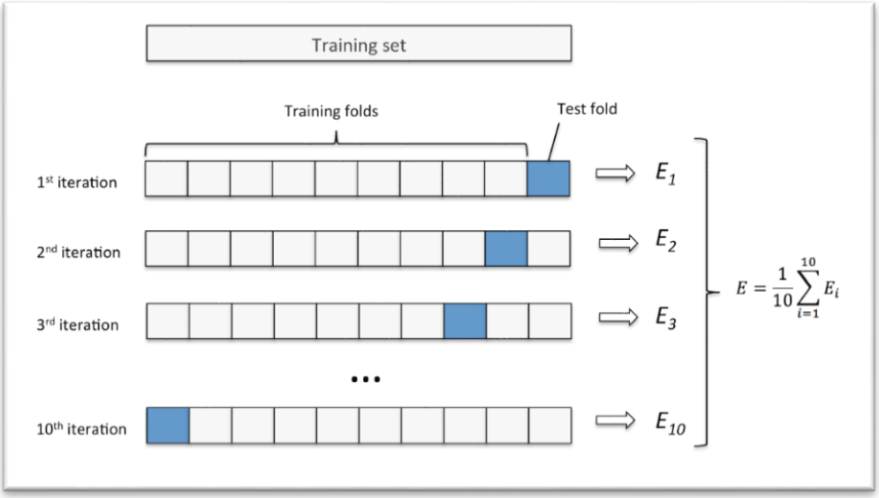
Cross-Validation

Aquí s'introdueix un nou concepte, la **validació** del model durant el procés d'entrenament (veure Il·lustració 7). Una de les tècniques més utilitzades és el *Cross-Validation* (CV), en el qual el *Train Set* es divideix de manera aleatòria en *n* grups, és a dir, en **K *Folds***, on K = *n*.

D'aquests grups, tots excepte un (K-1) s'utilitzen per entrenar el model, i el restant simula un *Test Set*, aquesta tasca es repeteix K vegades fins a obtenir K resultats, dels quals es fa una mitjana i s'obté una **estimació del rendiment**.

Per exemple, si es decideix fer 10 grups (K = 10), es farien 10 iteracions, amb 9 grups per entrenar i 1 pel testeig; cada vegada amb aquest últim diferent.

En resum, el CV serveix per poder comprovar la *accuracy* d'un model. A continuació s'observa el funcionament del mètode:

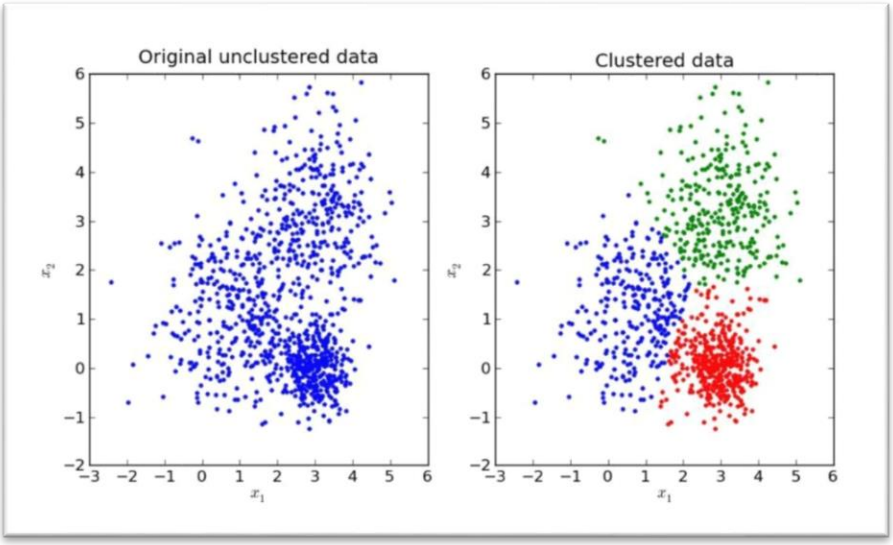


Il·lustració 7. K-Fold Cross-Validation
DataLab, "Cross Validation & Ensembling", 2020

2.1.1.2 Aprenentatge No Supervisat

L'aprenentatge no supervisat, a diferència de l'anterior, disposa només de variables d'entrada, sobre les quals intenta **establir patrons** i organitzar les dades segons variables en comú (veure Il·lustració 8).

Tot aquest procés es fa sense ajuda externa, és a dir, que és el propi model qui s'encarrega de trobar una estructura dins el conjunt, com es pot veure aquí:



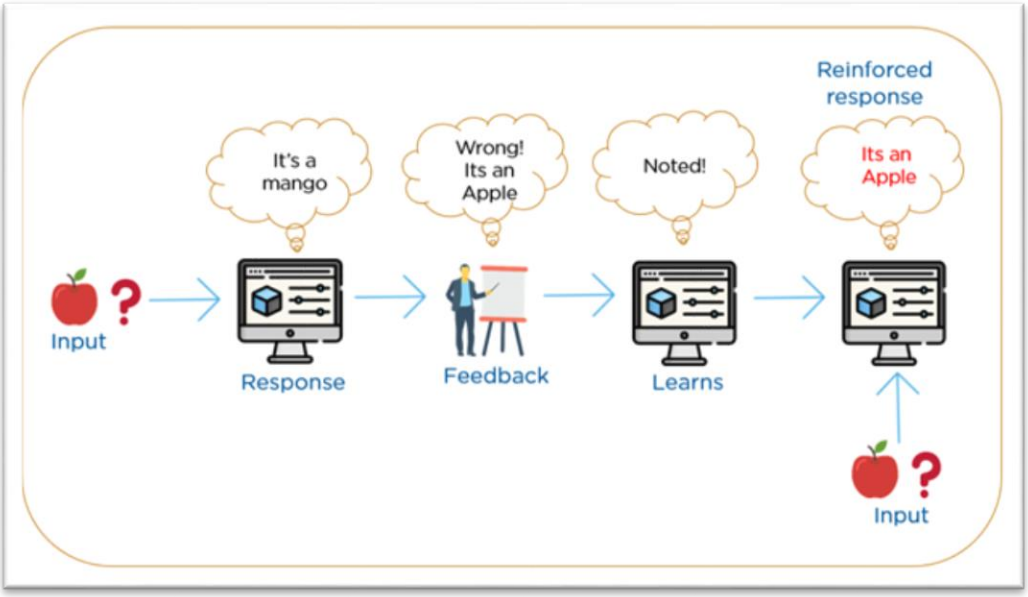
Il·lustració 8. Aprenentatge no supervisat
DeepAi, "What is Unsupervised Learning?"

Aquests algorismes tendeixen a ser més complicats, degut a que no disposen de tanta informació sobre les dades, per això, es basen en la cerca d'**exemples existents** amb valors comuns per classificar noves mostres.

2.1.1.3 Aprenentatge reforçat

L'aprenentatge reforçat podria considerar-se el més similar a la forma d'aprendre dels humans, això és perquè aprèn del seu entorn a mesura que hi va **interactuant** amb ell.

La idea principal és que el model va portant a terme vàries **accions** de les quals obté un resultat positiu o negatiu, el qual va sumant el seu coneixement (veure Il·lustració 9). Un exemple del seu funcionament seria el següent:



Il·lustració 9. Exemple d'aprenentatge reforçat
Shanika Perera, "An Introduction to Reinforcement Learning", 2019

Es veu com el model va provant diferents propostes fins que troba la correcta. Per aquesta naturalesa, s'utilitza en disciplines com la **teoria de jocs, de control, recerca, sistemes multi-agents**, entre d'altres.

Per al desenvolupament del projecte, la tècnica que s'ha triat ha sigut el **aprenentatge supervisat**, ja que es disposa d'una base de dades completa que inclou tant variables d'entrada com de sortida.

2.1.2 Models de Machine Learning

Per fer prediccions mitjançant *Machine Learning* s'ha de disposar d'un model entrenat, que per fer-ho pot seguir diferents estratègies. A continuació s'expliquen els principals.

2.1.2.1 Arbres de decisió

Els arbres de decisió son un mètode d'aprenentatge supervisat, usats tant per classificació com per regressió, en camps com la recerca, planificació estratègica, i com aquest projecte, pel *Machine Learning*.

Es basen en un arbre de decisió com a **model predictiu**, que està format per:

- **Branques**: Representen les observacions fetes sobre les dades, és a dir, conjuncions o característiques que porten a un node o un altre.
- **Fulles o nodes**: Son el valor objectiu, per tant, representarien una classe o eriqueta.

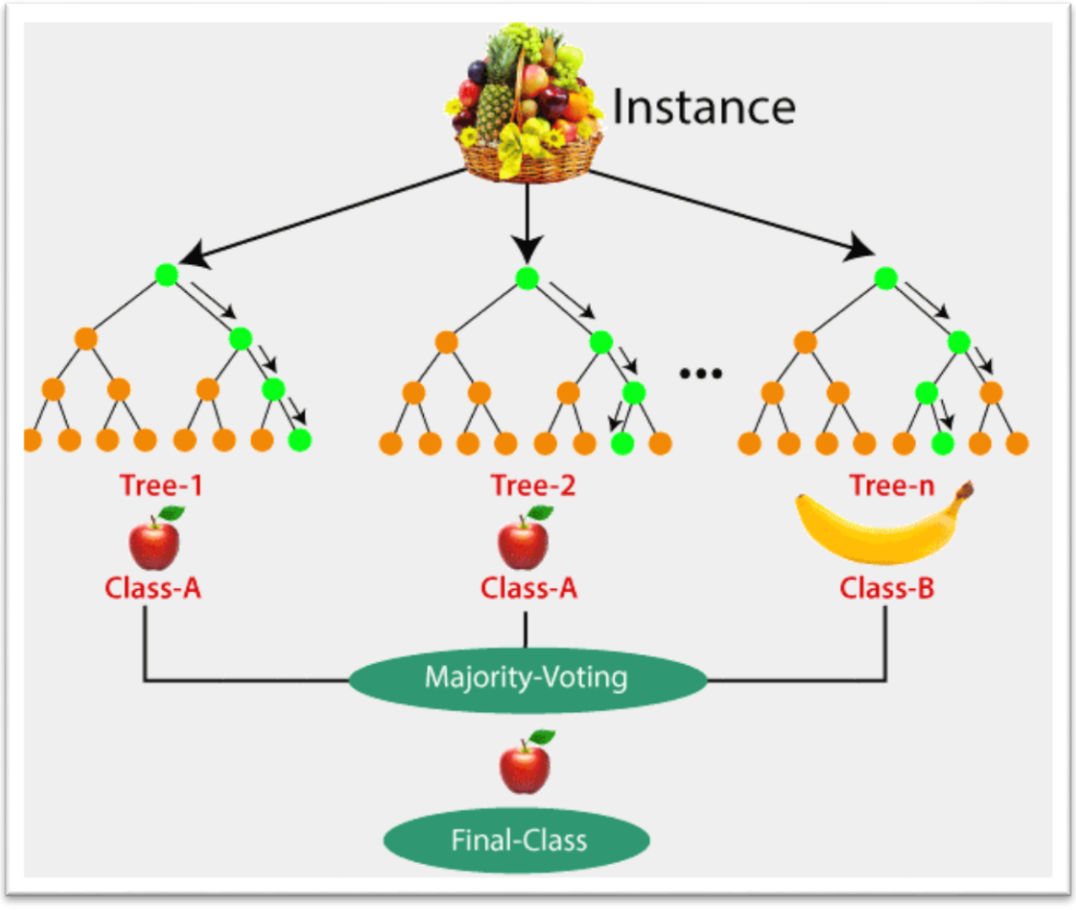
Tenen com a objectiu la creació d'un model per predir el valor o classificar una variable aprenent certes regles de decisió generades mitjançant el propi conjunt de dades.

Es podria dir que és el mètode més popular, ja que és molt fàcil d'interpretar i aprendre i el cost és força raonable, però al mateix temps pot portar a crear arbres massa complexos o ser sensible a petits canvis a les dades.

Random Forests

Quan fa una agrupació de diversos arbres de decisió per crear un model, s'obté un ***Random Forest***, o "Boscos aleatoris", que és una tècnica d'aprenentatge conjunta, ja que es basa en la creació de múltiples arbres pels quals passa les dades de forma aleatòria.

El resultat, que també pot ser una classificació, o un valor numèric, s'obté agafant el l'**opció majoritària** d'entre els arbres del model, reduint d'aquesta manera el risc d'error de cada arbre (veure Il·lustració 10).



Il·lustració 10. Funcionament d'un Random Forest
Section, "Introduction to Random Forest in Machine Learning", 2020

La imatge mostra en concret l'exemple d'un model intentant classificar una peça de fruita. Procedeix a preguntar a cada arbre de decisió que el forma, i com que la majoria (2/3) la defineix com a "Poma", el resultat final és **Poma**.

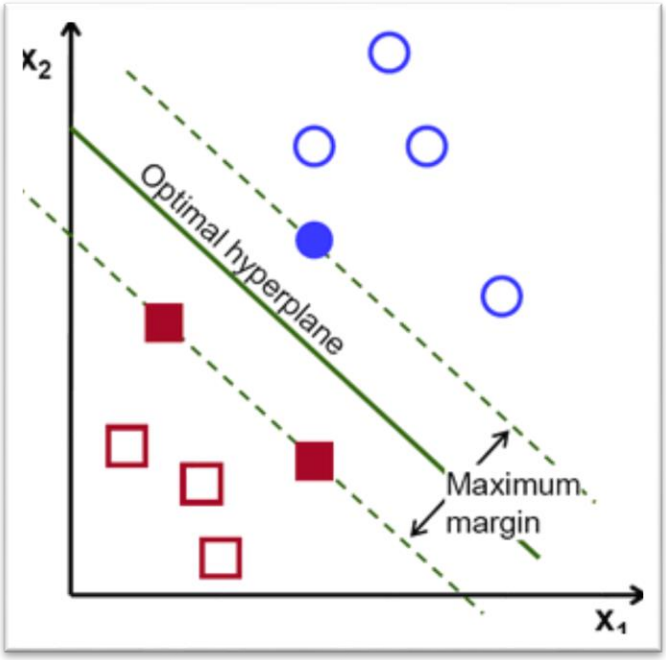
La imatge representa un bosc format per tres arbres de decisió individuals, als quals fa la mateixa "pregunta" i n'obté tres respostes diferents. I com es pot veure, la decisió final és la majoritària.

2.1.2.2 Support-Vector Machines

Els *Support-Vector Machines* (SVMs) també son un mètode d'aprenentatge supervisat, i igual que l'anterior, s'utilitzen tant per classificació com per regressió.

El seu funcionament parteix de la **detecció de *outliers***, és a dir, de valors atípics, que després utilitza per classificar dades entre dues classes, es tracta doncs d'un algorisme no probabilístic i **binari**.

És una tècnica (veure Il·lustració 11) que podria ser poc intuïtiva, a continuació hi ha un exemple que podria ajudar a entendre'l: suposem que disposem de dues classes "A" i "B" repartides en un pla; el model tractarà de trobar un hiperplà, és a dir un **vector** que separi les dues classes i estableixi un límit. Ho podem veure a la següent imatge:



Il·lustració 11. Funcionament del SVMs
Towards Data Science, "All Machine Learning Models Explained", 2020

Aquest vector final s'obté provant diferents direccions i trobant la distància mínima entre una mostra de cadascuna de les classes. Aquest hiperplà el converteix en un algorisme lineal, tot i així, és capaç de fer classificacions no lineals traçant els valors sobre espais de grans dimensions (anomenats *Kernels*).

2.1.2.3 Xarxes neuronals

També conegudes com a *Artificial Neural Networks* (ANN), aquesta tècnica consisteix en sistemes que imiten el **cervell** dels animals.

A diferència de les altres, no està específicament programada per fer prediccions, sinó que fa servir un aprenentatge basat en **exemples**, quants més en tingui més divers serà.

Es poden distingir tres capes:

- Capa d'entrada (***input layer***): fa referència a les variables d'entrada, com serien els exemples.
- Capa oculta (***hidden layer***): és la capa on el model transforma les entrades.
- Capa de sortida (***output layer***): la capa final, on el model retorna el seu resultat.

Aquestes capes estan formades per nodes s'envien **senyals** entre ells per compartir informació als nodes veïns, similar al funcionament de les neurones.

2.1.2.4 Xarxes Bayesianes

Les xarxes Bayesianes son un model **probabilístic** que utilitza la inferència Bayesiana per computar probabilitats. Això vol dir que representa la independència condicional d'un grup de variables aleatòries del conjunt de dades.

Al definir aquestes relacions és capaç de predir quines característiques podrien haver causat una situació. Per exemple, si tenim dues variables: "pluja" i "ruixador"; veiem que independentment no tenen **cap relació**, però si hi afegim "herba molla", immediatament hi apareix una relació degut a un **resultat comú**.

Aquest tipus de model es fa servir varis camps, com ara el dels diagnòstics, visió automatitzada, raonament, prediccions, presa de decisions en condicions d'incertesa i detecció d'anomalies.

2.1.2.5 Algorismes genètics

Els algorismes genètics (GA), son algorismes de cerca heurístics que es van basar en la teoria de la **selecció natural** de Charles Darwin, i degut a això tenen un funcionament similar.

El seu procés conté cinc fases: població inicial, funció d'adequació, selecció, encreuament i mutació. Aquestes fases s'alineen de la següent manera: de la població inicial es busquen els **individus en millor forma**, que son seleccionats per passar a la següent etapa on seran **encreuats** amb altres individus i s'obtindrà una **mutació superior**.

El seu ús es més limitat que la resta, ja que s'utilitzen sobretot per trobar solucions a problemes d'optimització.

2.1.3 Mètriques d'avaluació

Per a avaluar i comparar models, hi ha diverses mètriques d'avaluació, però la més utilitzada és la **matriu de confusió**, en la qual s'han basat les avaluacions d'aquest projecte.

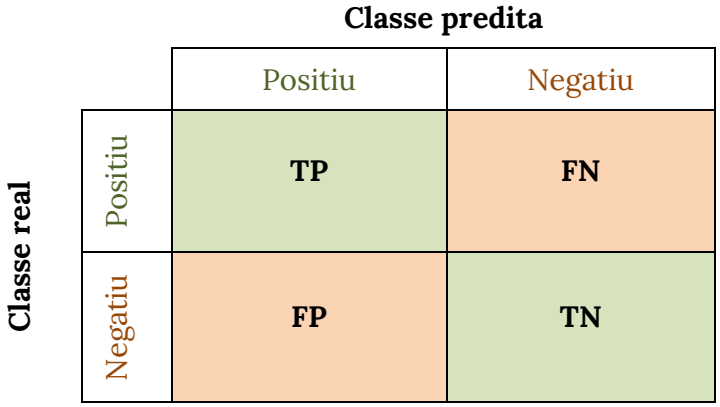
Matriu de confusió

Les matrius de confusió permeten visualitzar l'actuació dels models **d'aprenentatge supervisat** a l'hora de fer prediccions. Quan es tracta d'aprenentatge no supervisat

es parla de *Matching Matrix* (matriu d'emparellament), però no es tractaran durant el projecte ja que es treballarà amb algorismes d'aprenentatge supervisat.

El que permeten identificar aquestes matrius és quant i on ha encertat i s'ha equivocat un model, és a dir, mostren amb quines classes s'ha confós (si ha tingut errors) al donar un resultat.

Com indica el seu nom, son representades en forma de matriu, com l'exemple següent, on es diferencien dues classes:



La terminologia és la següent:

- **True Positive (TP):** Nombre d'encerts, classificacions fetes pel model on la classe predita és la classe real.
- **True Negative (TN):** Nombre de rebuigs encertats, les variables que el model ha classificat com a "no pertanyents" a la classe, realment no hi pertanyen.
- **False Positive (FP):** Nombre d'errors on el model ha classificat una variable com a part de la classe i realment no ho és.
- **False Negative (FN):** Nombre d'errors on el model classifica un model com a "no pertanyent" a una classe i realment si hi pertany.

Les mètriques que es poden obtenir a partir de la matriu de confusió son vàries, com ara les més bàsiques:

- *Accuracy*
- Sensibilitat (*True Positive Rate*)
- Especificitat (*True Negative Rate*)
- Precisió (*Positve Predicte Value*)
- *Negative Predictive Value*
- Taxa d'error

Les que s'han fet servir per comprovar els models, han sigut:

Accuracy

L'*accuracy* (exactitud) fa referència a la proximitat entre el valor predit i el valor real; és important no confondre-la amb la precisió, explicada a continuació.

L'exactitud es calcula com a la proporció de prediccions encertades, amb la següent fórmula:

$$\mathbf{ACC} = \frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN}$$

Precisió

La precisió, també anomenada "*Positive predictive value*", mesura la proximitat entre les mesures predites, com menor sigui la distància, major la precisió.

Es calcula com la proporció entre els encerts positius i el nombre de valors reals no pertanyents a una classe però classificats com a tal, d'aquesta manera:

$$\mathbf{PPV} = \frac{TP}{TP+FP}$$

Threat Score

El *Threat Score* (mesura de risc) o *Critical Success Index* (CSI), és una mètrica utilitzada per calcular l'anomenat *Intersection Over Union* (IoU), més conegut com a **Índex de Jaccard**, que mesura el grau de similitud entre dos conjunts.

Es calcula com a la proporció entre els encerts positius i aquests més tots els desencerts (negatius i positius):

$$\mathbf{TS} = \frac{TP}{TP+FN+FP} = \frac{Area\ of\ overlap}{Area\ of\ union} = \mathbf{IoU}$$

En resum, permet conèixer quant s'ha apropat la predicció del model al valor real.

2.1.4 AI en la medicina

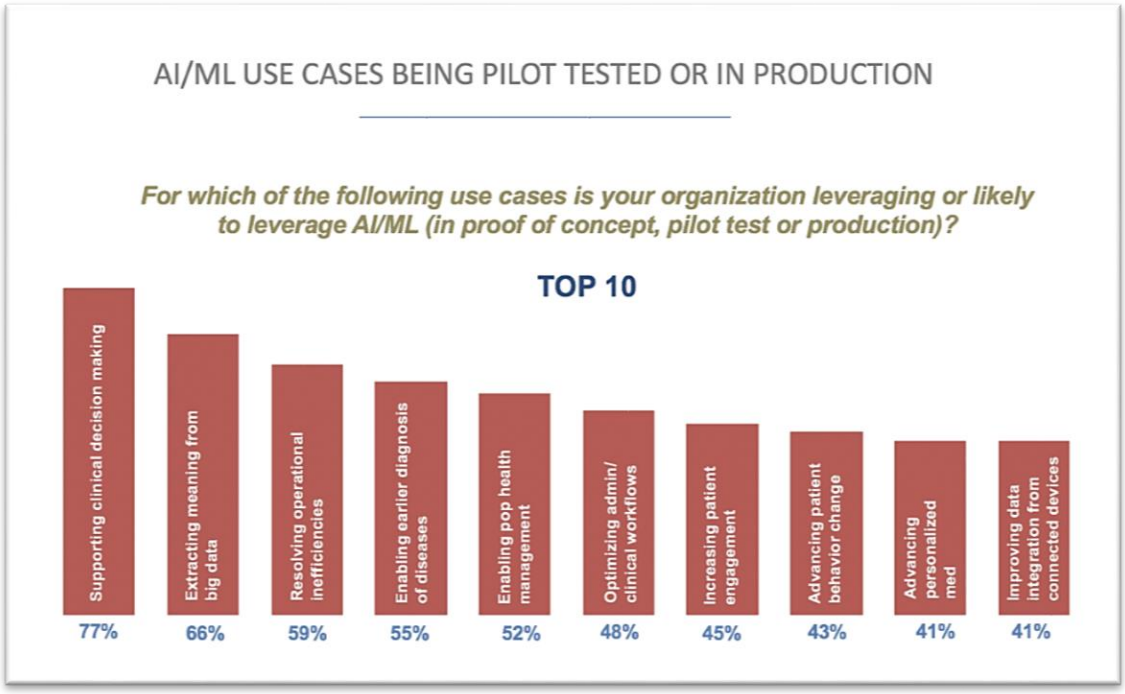
La Intel·ligència Artificial ha estat força present en l'àmbit sanitari durant aquests darrers anys, i ho està cada vegada més gràcies als beneficis que presenta i al gran potencial que té.

La IA en el camp de la medicina fa referència als algorismes dissenyats per realitzar tasques automatitzades sobre dades clíniques, que també son analitzen per poder suggerir **solucions a problemes mèdics**.

Actualment, algunes de les aplicacions de la IA son les següents:

- **Diagnosis mèdica:** s'utilitzen aquests mètodes per diagnosticar pacients amb malalties específiques.
- **Descobriment de fàrmacs:** sobretot en la indústria farmacèutica s'utilitza per dissenyar fàrmacs mitjançant l'estudi d'ingredients.
- **Assaigs clínics**: s'usen per estudiar progrés dels assaigs i recol·lectar dades sobre ells.
- **Algologia:** combinant la IA amb realitat virtual s'aconsegueix distraure al pacient i alleugerar els símptomes.
- **Cirurgies assistides:** permeten pre-analitzar les dades dels pacients per guiar al cirurgià durant les operacions i informar de possibles tècniques.
- **Anàlisi d'imatge:** per analitzar i diagnosticar en profunditat imatges obtingudes, per exemple, de radiografies.

La presa de decisions és l'ús principal d'aquesta tecnologia, com es pot veure en un estudi realitzat per ***Healthcare IT News***:



Il·lustració 12. Percentatges d'ús de la IA
Healthcare IT News, "3 charts show where AI is making an impact in healthcare", 2018

Aquestes son només un grup d'aplicacions (veure Il·lustració 12) de totes les possibilitats que ofereix la IA, però el que es busca realment és:

1. **Suport a l'hora de prendre decisions:** Poden prendre decisions més ràpides gràcies a la capacitat d'anàlisi dels algorismes.
2. **Administració de la informació:** Permet als metges automatitzar el procés de enregistraments i estudiar l'historial mèdic del pacient amb rapidesa.

A més, ha tingut també un gran impacte econòmic, ajudant a estalviar grans quantitats de diners a les autoritats i sobretot, vides.

L'objectiu principal és doncs, ajudar als treballadors sanitaris prendre millors decisions basades en el reconeixement de patrons i prevenir d'aquesta manera possibles complicacions de salut. I és també, un dels objectius d'aquest projecte.

2.2 Conceptes mèdics i la malaltia de la COVID-19

La malaltia del coronavirus 2019 (**COVID-19**), també identificada com a SARS-CoV-2 (*Severe Acute Respiratory Syndrome*), és una **malaltia respiratòria infecciosa**, que es va detectar per primer cop al desembre del 2019 a la ciutat xinesa de Wuhan.

Es tracta d'un virus provinent de la família dels coronavirus que afecten tant a humans com a animals, i és considerada una infermetat zoonòtica, ja que pot ser transmesa d'animals a humans. Actualment es desconeix el seu reservori natural o l'animal que ha pogut ser l'intermediari i possible transmissor.

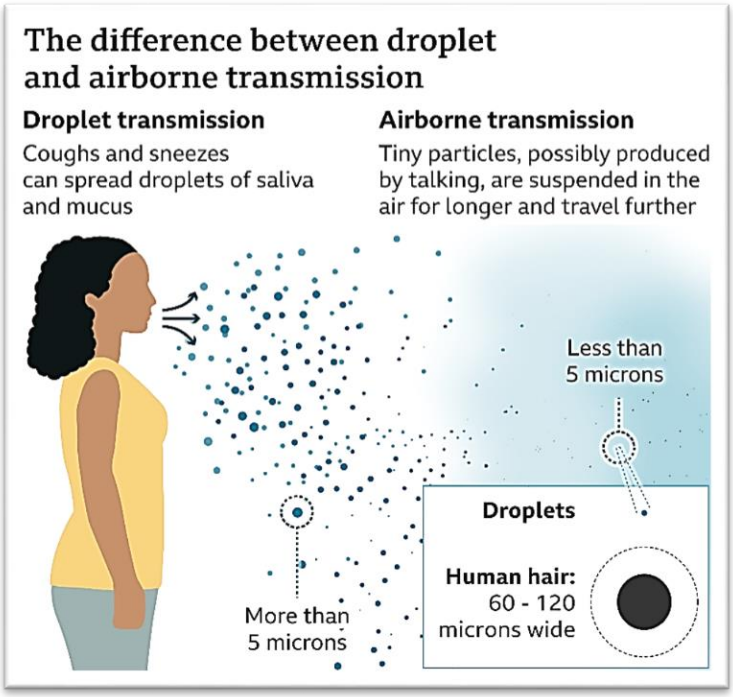
La Organització Mundial de la Salud (OMS) la va declarar com a **pandèmia** el 11 de març del 2020, i fins la data d'aquest projecte ha afectat a poc més de 174.600.000 persones i causat la mort de més de 3.700.000 persones.

Aquest virus té una certa complexitat, ja que pot arribar a tardar fins a **14 dies en mostrar símptomes** en un individu infectat, període en el qual hi existeix un risc de contagi. Un cop apareix la simptomatologia, aquests poden variar de persona en persona, però els més comuns son:

- Mal de cap i/o gola.
- Pèrdua d'olor i/o sabor.
- Congestió i secreció nasal.
- Tos.
- Dolor muscular.
- Febre.
- Dificultat respiratòria.

Tot i causar diversos símptomes hi ha casos de pacients que no han mostrat cap, donant-se sobretot en nens i la població jove.

La transmissió ocorre quan els individus tenen contacte amb les gotes respiratòries o **goteta de Flügge** que contenen el virus (veure Il·lustració 13), que podrien haver sigut exhalades per persones contagiades.



Il·lustració 13. Transmissió aèria de la COVID-19
OMS, "Difference between droplet and airborne transmission", 2020

El risc d'infecció es considera alt quan hi ha proximitat entre persones durant un llarg període de temps, és per això que una de les mesures preventives inclou el distanciament social. Tot i que el ritme de vacunació actualment està avançant, encara es mantenen les principals mesures de seguretat:

- Distanciament social.
- Ventilació d'espais tancats.

- Evitar llocs amb agrupacions de gent.
- Neteja de mans i bona higiene general.
- Netejar i desinfectar superfícies.
- Auto aïllament.

Curs clínic de la malaltia

Les característiques clíniques dels individus infectats han estat fins ara força similars, això s'ha pogut observar gràcies a la capacitat de diagnosis i la vigilància durant les diferents etapes de la pandèmia.

Aquestes característiques son vitals per a una possible predicció de l'estat d'un pacient, ja que proporcionaran al model una base d'aprenentatge. Algunes variables comunes a l'hora de l'ingrés son les següents: **febre, tos, mal de gola, vòmits, mareigs**, entre d'altres.

Els pacients, segons els seus símptomes, poden passar la malaltia lleu i ser demanats fer una quarantena al domicili, en canvi, si els símptomes son més greus poden ser ingressats a la UCI, o fins i tot, causar-los la mort.

Per aquesta raó, les constants a les etapes inicials i la informació com l'edat o el sexe, son valors potencials que ajudaran a fer una possible classificació de casos.

3 Estudi de viabilitat, metodologia i requisits

3.1 Estudi de viabilitat

En aquest apartat s'analitzarà la viabilitat del projecte, basant-se en sis punts: viabilitat econòmica, operacional, tècnica, de dades, legal i contractual i política.

Viabilitat econòmica

Una manera de calcular la viabilitat econòmica és mesurant els costos i els beneficis del projecte i valorar si aquets últims aporten suficient valor.

Comencem pels costos, que com es tracta d'un treball de desenvolupament d'un model, els associarem amb la part inicial d'un projecte: compres de hardware/software, formacions i dissenys. En aquest cas, ja es disposa d'ordinadors per portar-lo a terme, i la formació personal no suposa cap inversió monetària. Per tant, els costos els podem descartar.

Pel que fa als beneficis, diferenciem entre tangibles i intangibles, aquests primers no es tindran en compte, ja que el projecte no preveu comercialitzar-se, i els intangibles, no quantificables, busquen aportar una millora a la comunitat mèdica.

En conclusió, davant d'uns costos (gairebé) nuls i uns beneficis que proporcionen una ajuda als sanitaris, el projecte és **viable econòmicament**.

Viabilitat operacional

Ens concentrarem en respondre dues preguntes: "Val la pena solucionar el problema?" i "Quina és l'opinió dels usuaris?".

La primera es pot justificar mitjançant la importància d'un diagnòstic ràpid i la necessitat d'una previsió de l'estat dels pacients per a poder organitzar els recursos disponibles de manera eficient. I per a la segona, el projecte ha tingut en compte l'opinió dels metges i les seves preferències sobre els valors que es volen predir.

D'aquesta manera, es pot considerar **viable operacionalment**.

Viabilitat tècnica

La viabilitat tècnica estudiarà si es possible desenvolupar el projecte amb els recursos i tecnologies disponibles.

El model necessitarà una sèrie de dades per entrenar-se, les quals han sigut proporcionades per una font fiable. A més es compta amb la tecnologia que permetrà generar-lo: ordinadors, programes informàtics i llibreries; que alhora verificaran l'exactitud de les previsions.

Per tant, és **viable tècnicament**.

Viabilitat de dates

El càlcul del temps de les tasques del projecte s'ha fet en setmanes, resultant en un total de 14 setmanes, que és aproximadament el temps del qual es disposa durant el segon semestre. D'aquesta manera, podem dir que és **viable temporalment**.

Viabilitat legal i contractual

La viabilitat legal verificarà que el projecte es porta a terme respectant totes les lleis. Es treballarà amb informació de pacients reals, però es tracta de dades anònimes i verificades i utilitzades amb el permís de l'organització que ens les ha proporcionat: *HM Hospitales*.

Es també, per tant, **viable legalment**.

Viabilitat política

Finalment, la viabilitat política la podem verificar respecte a les normes de la pròpia universitat i específicament de la facultat, ja que es un treball individual i no per a una empresa. I com que es portarà a terme seguint el guió proporcionat als estudiants, es pot dir que és **viable políticament**.

Després d'haver comprovat els diferents punts, el resultat que s'ha obtingut confirma que el projecte **és viable**.

3.2 Metodologia

Per aquest projecte, que consisteix a crear un model de predicció, s'ha hagut de seguir una metodologia específica de desenvolupament de *software*, i de la mateixa manera, s'ha seguit una altra per documentar el seu procés de creació.

D'aquesta manera, es diferencien dues metodologies per cada apartat:

- Per a la memòria: **PMBOK** (*Project Management Body of Knowledge*).
- Per al desenvolupament del model: **Metodologia *Waterfall*.**

3.2.1 Metodologies seguides

3.2.1.1 PMBOK

Per a la documentació, s'ha fet servir una metodologia estàndard, basada en la guia escrita al *Project Management Body of Knowledge*, que detalla estàndards, pautes i normes per portar a terme la gestió de projectes.

Es basa en el treball fet seguint "bones pràctiques" i fa una organització en **processos** i **grups de processos** a seguir durant les diferents etapes d'un projecte. Aquests grups son:

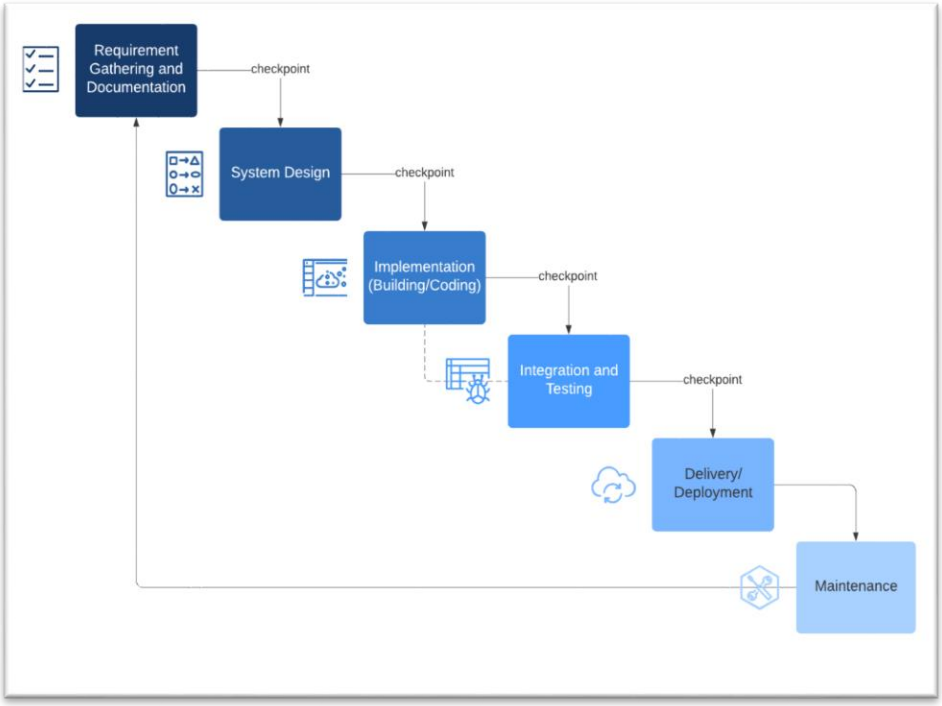
1. **Processos d'Inici:** és l'etapa on s'han començat a definir les idees pel projecte, s'ha estudiat el tipus de dades a treballar, quin resultat es volia obtenir i els possibles models.
2. **Processos de Planificació:** s'ha separat en tasques les parts del desenvolupament, com ho serien l'anàlisi de dades, la creació del model, el *testing* o el desplegament.
3. **Processos d'Execució:** la part principal del projecte, és on s'ha començat a treballar amb les dades per a crear el model.
4. **Processos de Monitoratge i Control:** ha consistit en la part de *testing* del model, on s'ha estudiat l'exactitud dels resultats i els possibles canvis de l'estructura del model.
5. **Processos de Tancament:** serà la part final del projecte, on es procedirà a fer el seu desplegament per a l'ús dels usuaris.

L'aportació d'aquestes pautes al projecte ha ajudat a portar el procés de documentació de manera organitzada i estructurada, i sobretot, a tenir una visió del temps disponible per gestionar-lo de manera correcta.

3.2.1.2 Waterfall

La creació del model, ha comptat amb una restricció: s'havia de completar una fase per poder passar a la següent, per això, s'ha seguit la metodologia anomenada *Waterfall* (veure Il·lustració 14), o desenvolupament en cascada, en la qual els projectes es divideixen en etapes linears, que depenen de l'anterior per a obtenir certes entrades.

El seu funcionament es podria resumir amb el següent esquema, que mostra acuradament el procés de creació d'un programa:



Il·lustració 14. Funcionament de la metodologia Waterfall. Lucidchart, "Waterfall Project Management Methodology"

Aquesta separació de fases permet deixar-les tancades una vegada realitzades, i son necessàries perquè proporcionen un punt de partida a la següent fase (veure Il·lustració 15). Un exemple del projecte seria el següent:

Il·lustració 15. Exemple d'entrades i sortides per a fases

A més, aquesta metodologia implica disposar d'una **documentació** important prèviament, ja que no permet desplaçar-se entre etapes, per això es disposa de la documentació de totes les llibreries fetes servir per a la creació del model i les dades que s'han analitzat, i també, ha requerit una etapa d'entrenament per conèixer l'àmbit de treball.

La decisió de triar aquesta metodologia ha permès veure una línia de temps clara, a on poder repartir les diferents tasques segons s'anaven obtenint resultats.

3.2.2 Etapes del projecte

Degut a la naturalesa de la metodologia triada, s'ha preparat el projecte agrupant-lo en etapes per tal d'assegurar que cada tasca estigués ben preparada i no requerís haver de tornar a una d'anterior. Aquestes etapes son, a grans trets:

- Aprenentatge i entrenament
- Desenvolupament
- Desplegament

3.2.2.1 Aprenentatge i entrenament

Aquesta fase ha consistit a preparar-se per a crear el model, per tant, el primer pas ha sigut entendre què és i perquè s'utilitza la Intel·ligència Artificial, específicament, *Machine Learning*, una branca seva.

Per preparar un bon model, s'han de conèixer els diferents mètodes de predicció existents, per avaluar quin d'ells s'adapta al resultat que es vol obtenir i quin ens serveix partint de les dades de les quals es disposen.

En resum, aquesta etapa ha donat com a sortida els mètodes triats, el coneixement i funcionament de les llibreries i una base per començar el desenvolupament.

3.2.2.2 Desenvolupament

La segona part, és el centre del projecte, la creació del model.

S'ha començat per l'anàlisi i normalització de la base de dades dels pacients, que ha proporcionat al programa informació neta i ordenada per poder ser entrenat. S'han provat diferents mètodes i tàctiques per obtenir la màxima exactitud, passant sempre per una fase de *testing*.

Aquesta etapa doncs, ha resultat en el model final, que consta de dues possibles prediccions: el estat futur dels pacients i els dies que podrà requerir ser ingressat a la UCI.

3.2.2.3 Desplegament

Finalment, un cop definit el model i passades totes les proves, s'ha passat a desplegar-lo en local, que ha consistit a crear una aplicació web que permet triar quin tipus de predicció es vol fer, entrar les dades del pacient i obtenir un resultat.

3.3 Requisits del sistema

El model del projecte, que servirà per fer previsions mèdiques, té com a objectiu que sigui utilitzat per metges o analistes. Es tractarà d'una aplicació web en la qual interactuaran diversos usuaris, assegurant sempre el seu bon funcionament. Per això, s'estudia en aquest apartat quines necessitats ha de satisfer i a qui ha de donar servei.

D'aquesta manera, els actors i rols als quals dóna lloc el disseny son:

- **Desenvolupador:** És qui crearà el model de predicció, i s'encarregarà de recol·lectar informació i netejar-la per entrenar el model.
- **Tester:** En aquest cas, serà el mateix desenvolupador, que verificarà que el model creat funciona correctament mitjançant dades de les quals es disposa el resultat.
- **Eina de predicció**: Internament, analitzarà les dades utilitzant llibreries per obtenir una predicció a través de les variables.
- **Client:** Usuari que farà servir el programa, proporcionarà certes variables d'entrada per obtenir la predicció.

Per tal que puguin desenvolupar el seu treball, el sistema haurà de satisfer una sèrie de requisits, funcionals i no funcionals, classificats per importància de la següent manera:

- Prioritat alta
- Prioritat mitja
- Prioritat baixa

3.3.1 Requisits funcionals

3.3.1.1 Desenvolupador

- **RFD1.** El desenvolupador ha de recopilar tota la informació necessària per a entrenar el model de predicció.
- **RFD2.** El desenvolupador ha de netejar i normalitzar les dades recopilades per evitar valors redundants o buits.

- **RFD3.** El desenvolupador ha de familiaritzar-se i entendre la informació d'entrada amb l'objectiu d'ordenar-la i organitzar-la per decidir quines variables estudiarà el model.
- **RFD4.** El desenvolupador ha de recopilar tota la informació necessària per a entrenar el model de predicció.
- **RFD5.** El desenvolupador ha de decidir quin(s) algorisme(s) i mètode(s) farà servir per a crear el model.
- **RFD6.** El desenvolupador crearà i entrenarà el model amb les dades netejades.
- **RFD7.** El desenvolupador cercarà informació i dades noves per poder actualitzar i millorar el model.
- **RFD8.** El desenvolupador desplegarà el model per al seu ús públic.

3.3.1.2 Tester

- **RFT1.** El tester ha d'obtenir dades amb resultats prèviament obtinguts per poder comparar-los amb els resultats de l'eina de predicció.
- **RFT2.** El tester ha d'utilitzar l'eina de predicció per verificar el correcte funcionament del programa.
- **RFT3.** El tester ha de portar un historial de prediccions correctes i incorrectes al llarg del desenvolupament del projecte.

3.3.1.3 Eina de predicció

- **RFE1.** L'eina ha de trobar característiques en comú entre les variables a analitzar i les dades amb les quals s'ha entrenat.
- **RFE2.** L'eina ha de fer una predicció partint de les característiques trobades en el punt anterior.
- **RFE3.** L'eina ha de mostrar el resultat de la predicció i el percentatge d'exactitud.
- **RFE4.** L'eina ha de permetre mostrar elements visuals d'ajuda (com gràfics o histogrames).

3.3.1.4 Client

- **RFC1.** El client ha de decidir quina o quines dades vol analitzar.
- **RFC2.** El client ha de poder proporcionar les variables d'entrada al model.
- **RFC3.** El client ha de poder triar quan iniciar l'anàlisi per a obtenir la predicció.
- **RFC4.** El client ha de poder observar el resultat de la predicció.

3.3.2 Requisits no funcionals

3.3.2.1 Usabilitat

- **RNFU1.** El programa ha d'explicar a l'usuari prèviament com utilitzar el model per a obtenir una predicció.

3.3.2.2 Eficiència

- **RNFE1.** El programa ha de donar una predicció en un temps limitat, passat aquest temps, ha d'informar a l'usuari.
- **RNFE2.** El programa ha de utilitzar les variables més útils per a fer la predicció, evitant l'augment del temps de resposta analitzant variables sense valor.

3.3.2.3 Confiabilitat

- **RNFC1.** El programa obtindrà el resultat més acurat possible, però sempre mostrant el percentatge d'error per deixar la decisió final en mans d'un professional.
- **RNFC2.** El programa ha de netejar les dades d'entrada, trobant i corregint valors nuls i normalitzar-los.
- **RNFC3.** El programa mostrarà transparència amb els models utilitzats per fer la predicció.
- **RNFC5.** En cas de fallada, el sistema intentarà tornar a executar el programa, i si torna a trobar un error, informarà a l'usuari, explicant la raó sempre que sigui possible.

4 Estudis, anàlisi i disseny

En aquest apartat es descriuran els detalls tècnics del projecte per entendre l'entorn i les condicions de treball abans d'explicar el desenvolupament. Per tant, inclou informació de la maquinària, llibreries i algorismes utilitzats, a més del disseny del model final.

4.1 Estudis i decisions

Per a crear un model de *Machine Learning* com el d'aquest projecte és vital partir d'una base de coneixements d'IA per poder analitzar les dades correctament i proporcionar al model els paràmetres necessaris, incloent l'algorisme d'aprenentatge.

Per aquesta raó, abans de començar el cos del treball, s'ha passat per un procés de familiarització en el qual s'han estudiat diferents mètodes d'aprenentatge automàtic en el llenguatge de programació ***Python***, l'escollit pel projecte (veure 3.1.1.2 Programari).

Inicialment, es va plantejar la creació de dos models:

- Un primer per la predicció de l'estat dels pacients contagiats, el model actual.
- I un altre per la diagnosi de la malaltia de COVID-19 amb anàlisi d'imatge mitjançant radiografies.

Amb aquesta idea, la preparació va constar en aprendre **processament d'imatges, normalització de dades** i mètodes de **predicció** i **classificació**.

L'estudi d'anàlisi d'imatge va incloure filtres com el *Gaussian*, *Median* i *Bilateral Filtering*, detecció de vores, la Transformada de Fourier i segmentació d'imatges.

Finalment, el tractament d'imatge no s'ha fet servir pel projecte, tot i així, la resta de mètodes apresos si constitueixen els algorismes utilitzats pel model, que seran descrits a continuació (veure 3.1.3 Algorismes).

Un cop finalitzat la fase d'entrenament previ, ja es podia començar el desenvolupament principal.

4.1.1 Entorn de treball

4.1.1.1 Maquinària

La totalitat del projecte s'ha realitzat en un mateix portàtil, amb les següents característiques:

HARDWARE

CPU	Processador Intel® Core™ i5-8250U, 1.60GHz, 2800Mhz.
Memòria RAM	8GB
GPU	NVIDIA GeForce MX130
Disc dur	256 GB SSD

SOFTWARE	
Sistema operatiu	Microsoft Windows 10 Home
IDE	Spyder 4.2.2
Software llenguatge de programació	Python 3.9.2
Disc dur	256 GB SSD

4.1.1.2 Programari i llenguatge

Python

Per codificar els models d'aquest projecte s'ha fet servir el llenguatge de programació ***Python*** (veure Il·lustració 16)**,** específicament la **versió 3.9.2**.

Es tracta d'un llenguatge interpretat i d'alt nivell amb la filosofia de que sigui entès per qualsevol persona amb coneixements bàsics de programació.

La seva organització el reconeix com a simple d'aprendre i multiparadigma, sobretot orientat a objectes, la qual cosa el fa flexible i adaptable.

Il·lustració 16. Icona Python, Python, 2008

La raó d'ús d'aquest llenguatge per al projecte és que suporta diferents mòduls i paquets i disposa d'una extensa varietat de llibreries que permeten el tractament de taules, imatges, anàlisi de dades, etc. I a més, és àmpliament utilitzat i recomanat en el sector de *Machine Learning*.

Spyder IDE

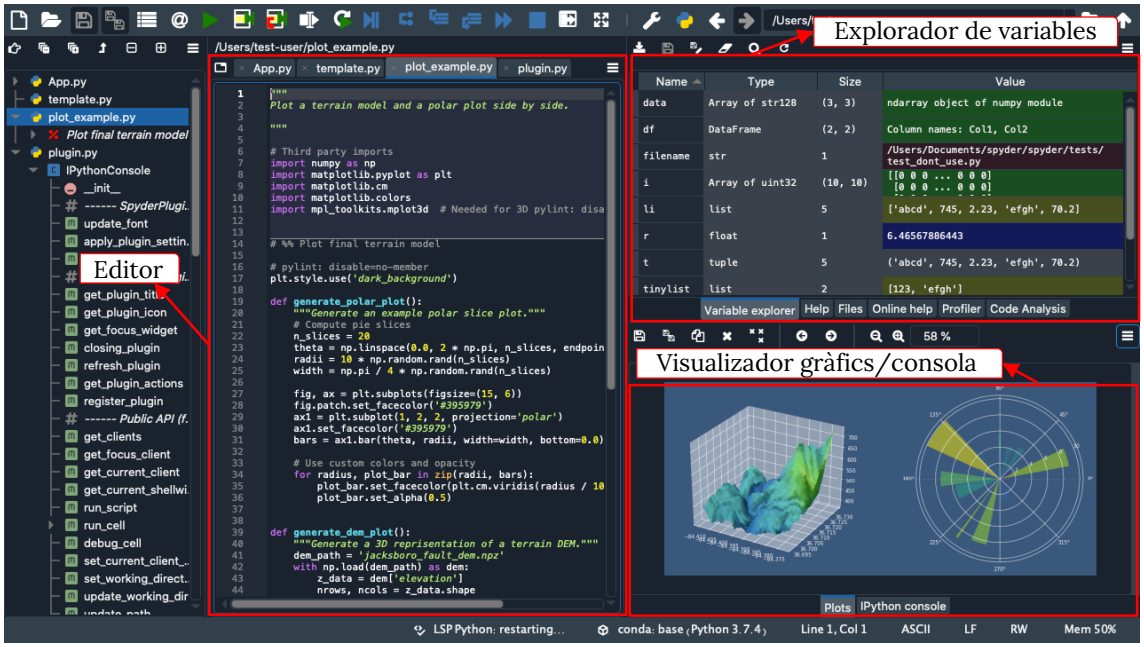
Spyder (veure Il·lustració 17) és l'**entorn de desenvolupament** utilitzat per portar a terme el projecte. És una plataforma creada en i per programar específicament en el llenguatge *Python*.

Està dissenyat especialment per a analistes de dades, científics i enginyers, i per això, compta amb característiques avançades d'edició, analítica i *debbuging*.

A més, està format per diversos components, com: l'editor, consola, explorador de variables i visualitzador de gràfiques; tots de ràpid accés al programar (veure Il·lustració 18)**.**

Il·lustració 17. Icona Spyder
Spyder IDE, 2019

Per entendre la seva organització, a continuació es mostra una captura de l'IDE amb els components principals:



Il·lustració 18. Components de Spyder IDE
Docs-spyder-ide, 2019

4.1.2 Llibreries utilitzades

Scikit-learn

Il·lustració 19. Icona Scikit Learn
Scikit Learn, 2018

Scikit-learn (veure Il·lustració 19) és una llibreria *open-source* creada per la programació d'aprenentatge automàtic en *Python*.

Disposa d'eines eficients i algorismes de classificació, regressió, màquines de vectors de suport, agrupacions, boscos aleatoris, *K-means*, i d'altres, que la converteix en un recurs complet per a crear un model de predicció.

Ofereix moltes funcions, algunes de les quals han servit pel projecte, com ara:

- Algorismes de aprenentatge supervisat.
- Algorismes de aprenentatge no supervisat.
- *Cross-Validation*.
- Extracció de característiques.

És important destacar que està enfocada a crear models de *Machine Learning* i no hauria de ser utilitzada per el tractament de taules o Data Sets.

Pel projecte, s'ha fet servir per cridar als algorismes de classificació, regressió i validació creuada.

A més, per tractar amb classificacions de classes no balancejades, s'ha optat per l'ús una llibreria basada en aquesta mateixa, **Imbalanced Learn** (veure Il·lustració 20), que proporciona eines per simular o disminuir dades.

Il·lustració 20. Icona Imbalanced Learn
Imbalanced Learn, 2021

Matplotlib

Matplotlib (veure Il·lustració 21) és una llibreria utilitzada per la **generació de gràfics** pel llenguatge *Python*. Aquests son creats mitjançant informació provinent de taules, conjunts de dades o llistes.

Es tracta també d'un codi *open-source*, que està dissenyat per tenir un funcionament similar al MATLAB i d'ús gratuït.

Il·lustració 21. Icona Matplotlib
Matplotlib, 2014

Al projecte s'ha utilitzat per a la visualització de gràfics, com histogrames, plànols de corbes i matrius de correlació.

Seaborn

Seaborn (veure Il·lustració 22) és també una llibreria de **visualització de dades** basada en l'anterior, Matplotlib. Permet generar plànols, corbes i gràfics estadístics amb una interfície més atractiva.

Il·lustració 22. Icona Seaborn
Seaborn, 2021

De manera similar a Matplotlib, s'ha fet servir per visualitzar matrius de correlació i recomptes de dades.

NumPy

NumPy (veure Il·lustració 23) és una llibreria d'alt nivell creada pel **tractament de vectors i matrius** multi-dimensionals en llenguatge *Python*. Proporciona també una varietat de funcions matemàtiques com la generació de nombres aleatoris o transformacions de Fourier.

Il·lustració 23. Icona NumPy
NumPy, 2009

S'ha fet servir per a la creació de vectors i tractament dels Data Sets.

Pandas

Pandas (veure Il·lustració 24) és també una llibreria *open-source* que dóna suport per a llistes com a extensió de la llibreria NumPy. Es tracta d'una eina ràpida i flexible de **manipulació i anàlisi de dades**, que proporciona estructures de dades i operacions per a tractar amb taules.

Il·lustració 24. Icona Pandas
pandas, 2016

La utilitat que ha tingut per a la creació del model ha sigut la de la càrrega de dades i estudi de taules i llistes.

Keras

Keras (veure Il·lustració 25) és una **API de xarxes neuronals** dissenyada per l'ús en l'aprenentatge profund. Ajuda a minimitzar les accions d'usuari comunes, reduint la càrrega cognitiva, i mostra missatges d'error per ajudar al programadors a localitzar problemes.

Il·lustració 25. Icona Keras
Keras, 2020

Tot i que al projecte no s'ha tractat el *Deep Learning*, si que s'ha utilitzat aquesta llibreria per a la funció de mesura denominada *Intersection Over Union*.

4.1.3 Algorismes

Per obtenir un bon model a l'hora de fer prediccions o classificacions, s'ha treballat amb tres algorismes de *Machine Learning* diferents per observar quin aconseguia els millors resultats partint de les mateixes entrades.

La base de dades disposa d'un nombre limitat de files, una mica més de 2000, que representen pacients contagiats, sense repeticions. Tot i ser una bona quantitat, els millors models d'aprenentatge automàtic requereixen informació molt més abundant i diferida per obtenir un coneixement profund.

El mètode que s'ha utilitzat pels dos models (predicció de l'estat futur del pacient i predicció de dies a la UCI), ha sigut el de **classificació**, per aquesta raó està format per algorismes d'aprenentatge **supervisat**, descrits a continuació, detallant la raó del seu ús.

Random Forests

Els boscos aleatoris (*Random Forests*) son un dels algorismes més utilitzats, tant per classificació com per regressió.

S'ha triat aquest algorisme com el principal, per les següents raons:

- Es consideren classificadors que poden aconseguir les ***accuracy*** més altes.
- Tenen un millor rendiment al tractar amb **classes no balancejades**, que és el cas de la base de dades de la qual es parteix.
- Les seves prediccions son les més **estables**.

Aquestes característiques venen donades sobretot per iterar les dades de forma aleatòria, que a més a més, el converteix en un classificador força objectiu i disminueix la dependència de dades.

Support Vector Machines

Les màquines de suport de vectors (SVM), també son algorismes que tenen utilitat per a la classificació i la regressió.

Estan pensats per a treballar amb dades que distingeixen **dues classes**, ja que és un sol vector el que estableix una frontera entre aquestes dues. Però, la llibreria Scikit-learn permet fer una classificació **multi-classe** amb un mètode lineal: ***Linear* SVC** (*Support Vector Classificator*). Aquest diferencia diferents classes amb la estratègia

coneguda com a *one-vs-the-rest* ("un contra la resta"), que compara cada classe amb totes les altres.

La raó del seu ús és doncs, a part de donar un competidor a l'algorisme anterior, que son també classificadors d'alta *accuracy* i eficients en quant a la seva **memòria** respecte les dades, proporcionada pels punts que generen els vectors durant el procés d'entrenament.

KNN

Finalment, s'ha optat per provar un altre algorisme: ***K-nearest neighbors*** (K veïns més propers). Igual que els anteriors, és apte tant per classificacions com per a la regressió.

Permet classificar els dades en més de dues classes, trobant els valors més propers entre ells per a separar-los de la resta, aconseguint resultats força similars als boscos aleatoris o als SVM. Però la principal raó de triar-lo ha sigut que treballen **millor amb conjunts de dades més petits**.

Publicació a una plataforma pública

Al finalitzar els models, s'ha decidit compartir-los pel ús predictiu, o per adaptar-lo a un futur projecte aliè a aquest. Per compartir el codi amb diferents usuaris, es farà servir la plataforma ***GitHub*** (veure Il·lustració 26)**,** un repositori en línia que permet guardar projectes i fer-los públics o d'ús privat.

Il·lustració 26. Icona GitHub
GitHub, 2008

4.2 Anàlisi i disseny del sistema

Es aquest apartat es plantejarà el disseny del sistema, prèviament s'han identificat els requisits del model, que serien les característiques que ha de complir (vegeu 2.3 Requisits del sistema). Aquests serviran de base per a entendre què farà exactament l'algorisme i com interactuarà amb els usuaris.

S'ha d'aclarir que el model no es troba en les etapes inicials i no està preparat per l'ús públic, es podria dir que és tracta d'una **primera versió**. Tot i així, com que s'ha creat amb l'objectiu de ser utilitzat per sanitaris, es proposaran dos escenaris:

- **Escenari 1 [Real]:** Els models podran ser descarregats i adaptats a les necessitats de l'usuari pel seu ús.
- **Escenari 2 [Ideal]:** Els models disposaran d'una interfície (aplicació web) on els usuaris podran fer les seves prediccions.

Escenari 1

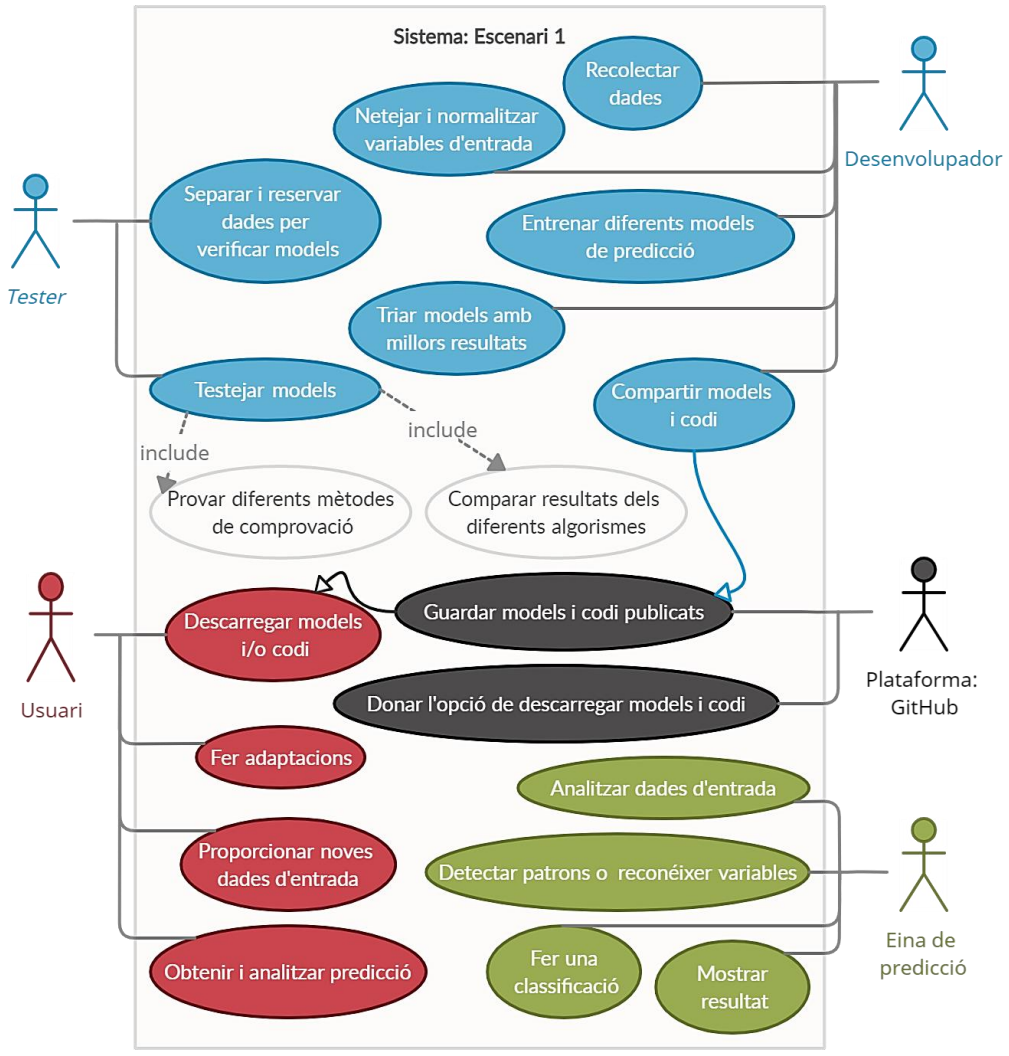
Aquest escenari (veure Il·lustració 27) mostra el funcionament del model actualment, és a dir, el que un usuari podrà fer quan finalitzi el projecte. L'idea a l'última fase és compartir el codi i els models entrenats a una plataforma pública a Internet, *GitHub*.

Per tant, queda una relació entre: usuaris, desenvolupador i *tester*, l'eina de predicció i la plataforma per a compartir el codi.

El funcionament serà el següent:

1. El desenvolupador entrena i comprova el funcionament del model.
2. El desenvolupador puja el model i el codi a una plataforma pública pel seu ús.
3. Un usuari extern descarrega el codi i els models, i si ho veu necessari, ho adapta les seves necessitats.
4. L'usuari proporciona unes dades d'entrada al model.
5. El model analitza les dades i fa una classificació.
6. El model mostra el resultat.

Aquesta situació resulta en el següent diagrama d'ús:



Il·lustració 27. Diagrama d'ús de l'escenari 1
Creat amb "Creately", 2021

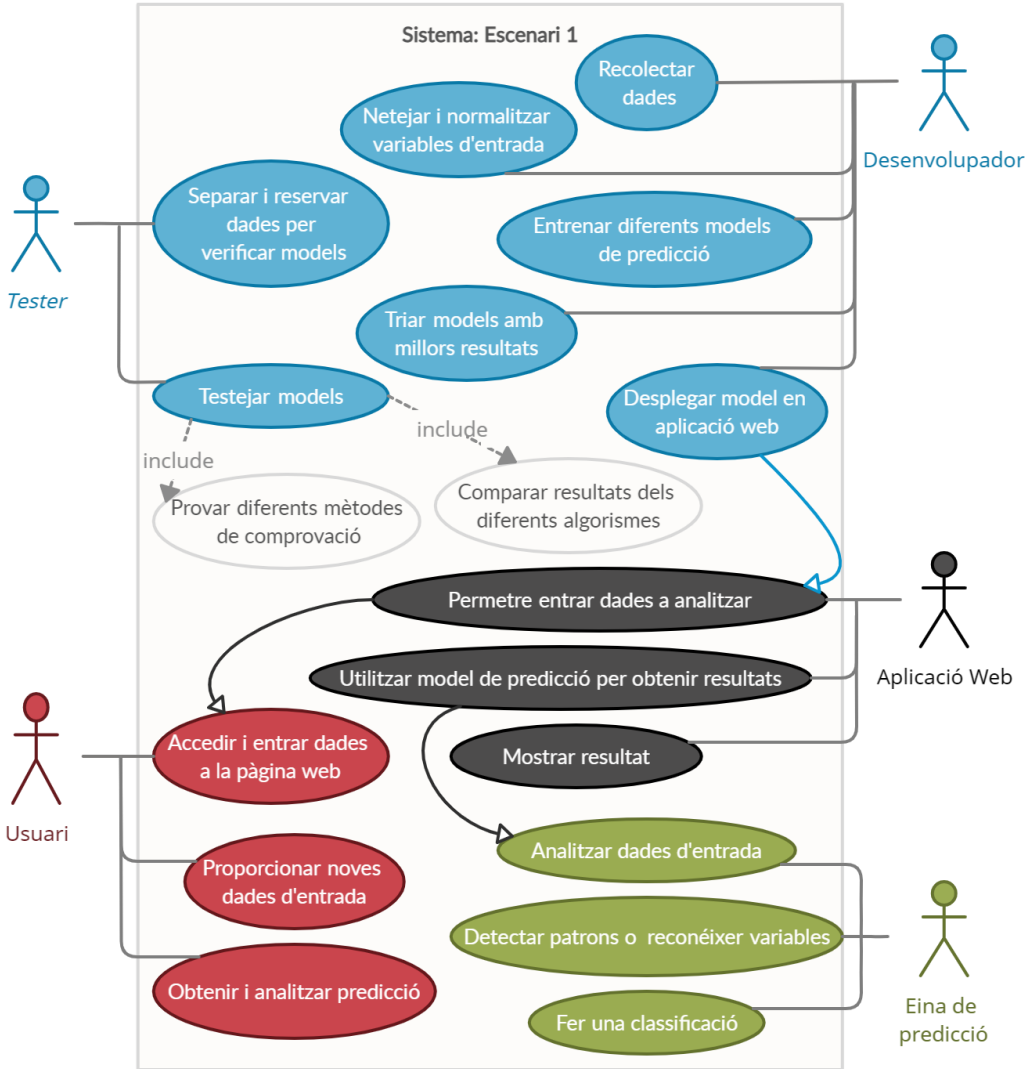
Escenari 2

Aquest escenari (veure Il·lustració 28) simularia un cas de desplegament real, un cop el model estigués totalment acabat i hagués passat un procés de proves intensives podria ser apte per l'ús professional.

Els participants son els mateixos que l'escenari real, excepte la plataforma pública, que seria intercanviada per una aplicació web on es farien les prediccions directament. Els passos serien els següents:

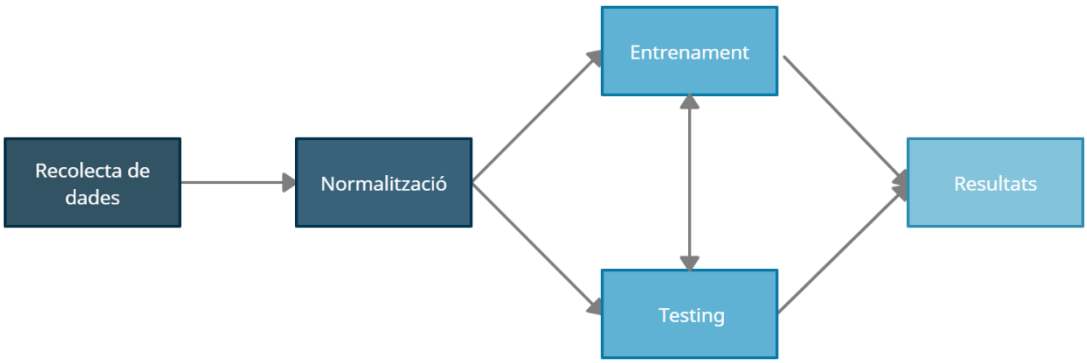
1. El desenvolupador entrena i comprova el funcionament del model.
2. El desenvolupador desplega el model a una aplicació web.
3. Un usuari accedeix a la pàgina web.
4. L'usuari escriu les dades d'entrada a la web.
5. L'aplicació traspassa les dades al model per ser analitzades.
6. L'aplicació mostra el resultat obtingut pel model.

Aquesta situació resulta en el següent diagrama d'ús:



Il·lustració 28. Diagrama d'ús de l'escenari 2
Creat amb "Creately", 2021

Ambdós escenaris coincideixen en la primera part de la interacció (veure Il·lustració 29): el desenvolupament i el *testing*, la part principal del projecte, que es pot esquematitzar de la següent manera:



Il·lustració 29. Disseny del procés de desenvolupament
Creat amb "Creately", 2021

Aquesta imatge representaria el disseny del procés de desenvolupament, on es comença obtenint les dades i normalitzant-les per entrenar el model. Un cop entrenat es passa a la fase de *testing*, on si els resultats no son prou convincents, es pot tornar a entrenar el model fins aconseguir uns resultats satisfactoris.

5 Implementació, proves i resultats

Fins ara s'han explicat els aspectes teòrics i antecedents per entendre per a què s'han creat els models i com s'han creat i avaluat. Aquest apartat es podria considerar el cos del projecte, on s'ha combinat el procés de **desenvolupament** del model i els **resultats**, degut a que son fases que estan fortament lligades.

A mode de resum, es poden trobar els següents punts:

- Normalització.
- Implementació.
- Comparació de diferents implementacions.
- Resultats finals.

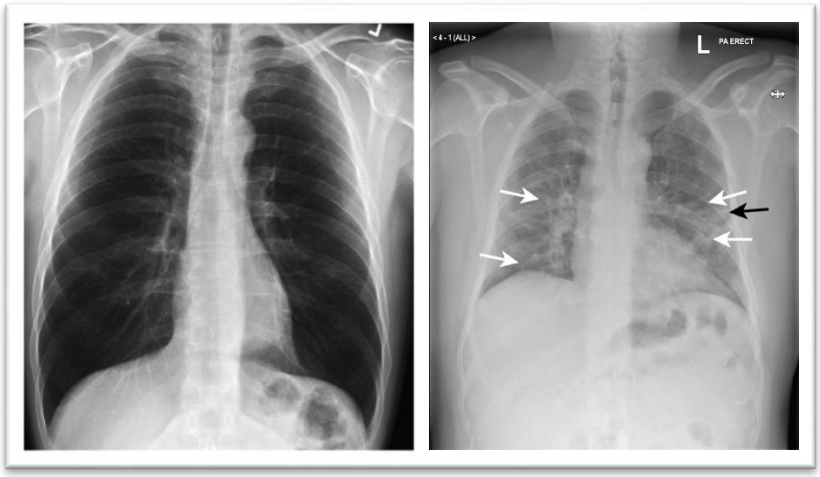
5.1 Implementació i proves

5.1.1 Anàlisi de la base de dades i normalització

Un dels objectius mencionats a la introducció era l'anàlisi i normalització de la base de dades, això permetrà familiaritzar-se amb les dades i reconèixer les variables clíniques per a adaptar el model.

Abans de aprofundir en el procés, repassem que conté la base de dades facilitada per *HM Hospitales*. Com s'ha mencionat a la introducció (veure 1.2 Base de dades), inclou dos tipus de dades:

1. **Imatges:** Conjunt de radiografies de tòrax de pacients de COVID-19 anonimitzades (veure Il·lustració 30).
2. **Taules de variables clíniques:** un total de sis taules organitzades segons el seu contingut, incloent les dades d'ingrés dels pacients, constants vitals i medicaments subministrats.



Il·lustració 30. Radiografia pulmons sans (esquerra) i pulmons d'un pacient de COVID-19[1]
Esquerra: Radiopaedia, "Normal chest x-ray"
Dreta: TheBMJ, "Role of chest radiography in confirmin Covid-19 pneumonia", 2020

[1] Per raons de protecció de dades no s'han compartit imatges reals de la base de dades.

Cap dels dos models que s'han creat fa anàlisi d'imatge, per tant, aquest conjunt de radiografies no s'ha fet servir. Per aquesta raó a continuació s'explicaran les taules que si s'han utilitzat i el seu contingut.

Taules de la base de dades

Per crear els models, s'ha de decidir primer quines variables clíniques serviran per a entrenar-los. Per això, el **primer pas** ha sigut observar el contingut de les taules.

Les sis taules que formen la base de dades s'han proporcionat en el format de fitxers "**.csv**" i contenen diferents quantitats d'informació sobre els pacients, per això, en aquest sub-apartat s'especificaran també el nombre de registres.

Taula 1: Taula principal

Aquesta taula conté dades de pacients com ara mesures preses durant l'ingrés, dades agafades durant l'última estada a les urgències i dades durant l'estada a la UCI, si n'hi ha hagut.

Conté **2307 registres** de diferents pacients amb aquestes columnes per a cadascun d'ells:

Columna	Significat
PATIENT ID	Identificador únic del pacient ingressat
EDAD/AGE	Edat del pacient a data d'abril del 2020
SEXO/SEX	Sexe del pacient
DIAG ING/INPAT	Diagnosis COVID a l'ingrés
F_INGRESO/ADMISSION_D_ING/ INPAT	Data d'ingrés com a pacient
F_ENTRADA_UC/ICU_DATE_IN	Data d'entrada a la UCI (si s'ha requerit)
F_SALIDA_UCI/ICU_DATE_OUT	Data de sortida de la UCI (si s'ha entrat)
UCI_DIAS/ICU_DAYS	Dies d'ingrés a la UCI (si s'ha requerit)
F_ALTA/DISCHARGE_DATE_ING	Data d'alta com a pacient ingressat
MOTIVO_ALTA/DESTINY_DISCHA RGE_ING	Raó o destí d'alta
F_INGRESO/ADMISSION_DATE_ URG/EMERG	Data de visita a urgències que va resultar en ingrés
HORA/TIME_ADMISION/ ADMISSION_URG/EMERG	Hora de visita a urgències que va resultar en ingrés
ESPECIALIDAD/DEPARTMENT_ URG/EMERG	Especialitat d'urgències que va atendre al pacient
DIAG_URG/EMERG	Diagnosis al moment d'admissió a urgències
DESTINO/DESTINY_URG_EMERG	Destí a urgències
HORA/TIME_CONSTANT_PRIMERA /FIRST_URG/EMERG	Hora del primer registre de constants agafades a urgències

TEMP_PRIMERA/FIRST_URG/ EMERG	Primer registre de temperatura agafat a urgències
FC/HR_PRIMERA/FIRST_URG/ EMERG	Primer registre de freqüència cardíaca agafat a urgències
GLU_PRIMERA/FIRST_URG/ EMERG	Primer registre de glucèmia agafat a urgències
SAT_02_PRIMERA/FIRST_URG/ EMERG	Primer registre de saturació d'oxigen agafat a urgències
TA_MAX_PRIMERA/FIRST/EMERG _URG	Primer registre de tensió arterial màxima agafat a urgències
TA_MIN_PRIMERA/FIRST_URG/ EMERG	Primer registre de tensió arterial mínima agafat a urgències
HORA/TIME_CONSTANT_ULTIMA /LAST_URG/EMERG	Hora de l'últim registre de constants agafats a urgències
FC/HR_ULTIMA/LAST_URG/ EMERG	Últim registre de freqüència cardíaca agafat a urgències
TEMP_ULTIMA/LAST_URG/ EMERG	Últim registre de temperatura agafat a urgències
GLU_ULTIMA/LAST_URG/EMERG	Últim registre de glucèmia agafat a urgències
SAT_02_ULTIMA/LAST_URG/ EMERG	Últim registre de saturació d'oxigen agafat a urgències
TA_MAX_ULTIMA/LAST_ URGEMERG	Últim registre de tensió arterial màxima agafat a urgències
TA_MIN_ULTIMA/LAST_URG/ EMERG	Últim registre de tensió arterial mínima cardíaca agafat a urgències

Taula 2: Taula de medicació

La segona taula conté dades sobre la medicació pautada dels pacients i la quantitat administrada durant el ingrés. Compta amb un total de **60.460 registres** de diferents dates i medicaments (els pacients apareixen més d'un cop, ja que se'ls ha subministrat més d'un fàrmac i en diferents dates). La formen aquestes columnes:

Columna	**Significat**
PATIENT ID	Identificador únic del pacient ingressat
FARMACO/DRUG_NOMBRE_ COMERCIAL/COMERCIAL_NAME	Nom comercial del fàrmac
DOSIS_MEDIA_DIARIA/DAILY_ AVRG_DOSE	Mitjana de la dosis diària subministrada
INICIO_TRAT/DRUG_START_DATE	Inici del tractament
FIN_TRAT/DRUG_END_DATE	Fi del tractament
ATC5_NOMBRE/NAME	Descripció de la categoria ATC5 en la classificació ATC
ID_ACT	Identificador de la categoria ATC5 en la classificació ATC

ATC7_NOMBRE/NAME	Descripció de la categoria ATC7 en la classificació ATC
ID_ACT7	Identificador de la categoria ATC7 en la classificació ATC

Taula 3: Taula de constants vitals

Aquesta taula conté els registres de les constants vitals preses durant el ingrés del pacient, no inclou les de la UCI. Amb un total de **55.515 registres** (també amb pacients repetits perquè conté diferents dades), té les següents columnes:

Columna	Significat
PATIENT ID	Identificador únic del pacient ingressat
CONSTANTS_ING/INPAT_FECHA/DATE	Data de registre de la constant presa
CONSTANTS_ING/INPAT_HORA/TIME	Hora de registre de la constant presa
FC/HR_ING/INPAT	Valor de la freqüència cardíaca
GLU/GLY_ING/INPAT	Valor de la glucèmia
SAT_02_ING/INPAT	Valor de la saturació d'oxigen
ID_ACTTA_MAX_ING/INPAT	Valor de la tensió arterial màxima
TA_MIN_ING/INPAT	Valor de la tensió arterial mínima
TEMP_ING/INPAT	Valor de la temperatura

Taula 4: Taula de laboratori

Aquesta taula inclou els registres dels resultats de les peticions de laboratori realitzades durant el ingrés dels pacients. Amb un total de **396.055 registres** (també amb pacients repetits), està organitzada en les següents columnes:

Columna	Significat
PATIENT ID	Identificador únic del pacient ingressat
PETICION_LABORATORIO/LAB_NUMBER	Identificador de la petició de laboratori
FECHA_PETICION/LAB_DATE	Data de la petició de laboratori
HORA_PETICION/TIME_LAB	Hora de la petició de laboratori
DETERMINACION/ITEM_LAB	Determinació
RESULTADO/VAL_RESULT	Resultat de la determinació
UNIDADES/UD_RESULT	Unitats de mesura del resultat de la determinació
VALORES_REFERENCIA/REF_VALUES	Valors de referència per a la determinació

Taula 5: Taula de codificacions CIE10

Aquesta última taula conté els registres de codificació segons la normativa de la CIE10 (Classificació Internacional d'Infermetats, 10ª edició), i està separada en dues[2]:

1. Registres de codificació d'urgències segons la CIE10: amb **1987 registres**.
2. Registres de codificació d'ingrés segons la CIE10: amb **1775 registres**.

Un cop conegut el contingut de cada taula es va decidir triar quina o quines serien les més apropiades per entrenar el model. El primer model té com a objectiu predir l'estat futur d'un pacient, mentre que el segon retornarà un rang dels dies que un pacient podria passar ingressat a la UCI.

Inicialment es van veure les taules següents com a potencials:

- Taula 1: imprescindible, ja que conté totes dues variables de sortida, l'estat futur del pacient (**raó d'alta**) i **dies a la UCI**, a més de dades bàsiques dels pacients.
- Taula 2: es va creure que els fàrmacs administrats podrien afectar a l'evolució dels pacients.
- Taula 3: les constants vitals també podrien ajudar a estudiar l'evolució.

I la resta es va decidir reservar-les per a futures millores del model.

Procés de normalització

A partir d'aquí comença la primera part de la generació de codi per entrenar el model. Per fer-ho s'han de preparar les dades, guardades en les taules mencionades a l'apartat anterior.

Tenir les dades preparades vol dir que aquestes han de ser rellevants, estar completes i equilibrades i no tenir camps buits. A aquest procés se li anomena **normalització**, que permet assegurar la integració de dades i disminució de la redundància.

Taula 1

Pas 1: Càrrega i anàlisi de la variable a predir

Un cop obert l'entorn de desenvolupament i carregades les llibreries necessàries (veure Il·lustració 31), el següent pas es **carregar la taula** i guardar-la en un *DataFrame* (veure Il·lustració 32 i 33).

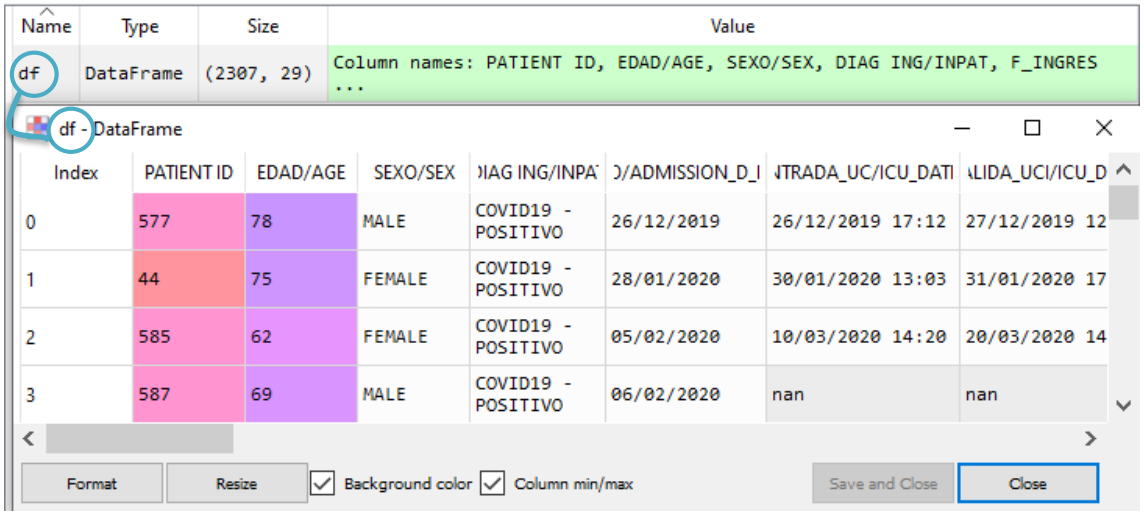
```
import pandas as pd
```

Il·lustració 31. Importació de la llibreria "pandas" utilitzada per a carregar la taula

```
df = pd.read_csv('../Database/HM_Hospitales/csv/01.csv')
```

Il·lustració 32. Comanda per a carregar la taula en un DataFrame

[2] Les variables que formen les dues taules estan especificades a l'ANNEX degut al gran nombre de columnes i que no s'han utilitzat duran el procés d'entrenament, per tant, es considera informació extra.



Il·lustració 33. Taula carregada com a DataFrame a l'explorador de variables

Els DataFrames permeten mostrar un resum (veure Il·lustració 34) del seu contingut, com ara el total de files, mitjanes, màxims, etc., però mostra només els valors numèrics. Al observar-la podem comprovar que realment hi ha **2307 registres**, és a dir, 2307 pacients diferents, i que per exemple, la mitjana de dies ingressats a la UCI son 7.

	count	mean	...	75%	max
PATIENT ID	2307.0	1161.774166	...	1741.5	2321.0
EDAD/AGE	2307.0	67.599046	...	80.0	106.0
UCI_DIAS/ICU_DAYS	201.0	7.139303	...	10.0	37.0
TEMP_PRIMERA/FIRST_URG/EMERG	2307.0	28.005028	...	36.9	40.1
FC/HR_PRIMERA/FIRST_URG/EMERG	2307.0	69.340269	...	97.0	190.0
GLU_PRIMERA/FIRST_URG/EMERG	2307.0	1.961855	...	0.0	448.0
SAT_02_PRIMERA/FIRST_URG/EMERG	2307.0	72.394452	...	96.0	99.0
TA_MAX_PRIMERA/FIRST/EMERG_URG	2307.0	83.437798	...	137.0	220.0
TA_MIN_PRIMERA/FIRST_URG/EMERG	2307.0	48.182488	...	79.0	845.0
FC/HR_ULTIMA/LAST_URG/EMERG	2307.0	73.991331	...	99.0	190.0
TEMP_ULTIMA/LAST_URG/EMERG	2307.0	29.063546	...	37.0	89.0
GLU_ULTIMA/LAST_URG/EMERG	2307.0	2.308192	...	0.0	448.0
SAT_02_ULTIMA/LAST_URG/EMERG	2307.0	77.016905	...	96.0	99.0
TA_MAX_ULTIMA/LAST_URGEMERG	2307.0	88.669701	...	138.5	220.0
TA_MIN_ULTIMA/LAST_URG/EMERG	2307.0	50.982228	...	80.0	845.0

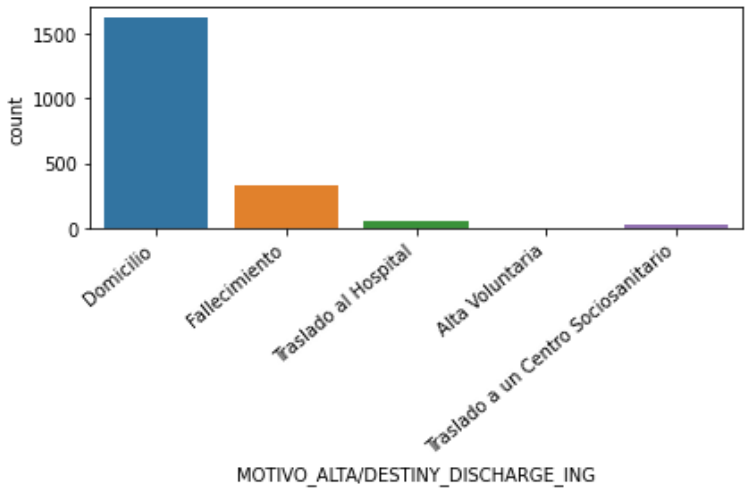
Il·lustració 34. Descripció d'un DataFrame amb la Taula 1

Una variable que aquí no es mostra, però que és molt important és la de la raó d'alta, és a dir, el valor que s'ha decidit que serà la sortida del model. Això és degut a que no és un valor numèric, sinó una cadena de caràcters.

Per veure una aproximació de les diferents sortides, s'ha fet servir la llibreria *Seaborn* (veure Il·lustració 35), que ha generat un histograma d'aquestes (veure Il·lustració 36).

```
import seaborn as sns
```

Il·lustració 35. Importació de la llibreria "Seaborn" per a generar gràfics



Il·lustració 36. Histograma dels valors de la columna "MOTIVO_ALTA"

El gràfic mostra que hi ha **cinc classes** diferents: "Domicilio", "Fallecimiento", "Traslado al Hospital", "Alta Voluntaria" i "Traslado a un Centro Sociosanitario". Entre aquestes seria on el model hauria de fer una **classificació** al rebre noves variables d'entrada.

Pas 2: Comprovar valors nuls

Hi ha dues maneres de tractar amb valors nuls: eliminar-los o emplenar-los (ja sigui amb mitjanes o valors simulats), però és important localitzar-los, ja que si el model se'n troba amb un d'ells no sabrà com classificar-lo. Per tant, el següent pas és mirar quants valors nuls té la taula (veure Il·lustració 37).

```
PATIENT ID                                    0
EDAD/AGE                                      0
SEXO/SEX                                      0
DIAG ING/INPAT                                0
F_INGRESO/ADMISSION_D_ING/INPAT               0
F_ENTRADA_UC/ICU_DATE_IN                   2088
F_SALIDA_UCI/ICU_DATE_OUT                  2106
UCI_DIAS/ICU_DAYS                          2106
F_ALTA/DISCHARGE_DATE_ING                   218
MOTIVO_ALTA/DESTINY_DISCHARGE_ING           259
F_INGRESO/ADMISSION_DATE_URG/EMERG           83
HORA/TIME_ADMISION/ADMISSION_URG/EMERG       83
ESPECIALIDAD/DEPARTMENT_URG/EMERG            83
DIAG_URG/EMERG                               83
DESTINO/DESTINY_URG/EMERG                    83
HORA/TIME_CONSTANT_PRIMERA/FIRST_URG/EMERG   457
TEMP_PRIMERA/FIRST_URG/EMERG                  0
FC/HR_PRIMERA/FIRST_URG/EMERG                 0
GLU_PRIMERA/FIRST_URG/EMERG                   0
SAT_02_PRIMERA/FIRST_URG/EMERG                0
TA_MAX_PRIMERA/FIRST/EMERG_URG                0
TA_MIN_PRIMERA/FIRST_URG/EMERG                0
HORA/TIME_CONSTANT_ULTIMA/LAST_URG/EMERG     83
FC/HR_ULTIMA/LAST_URG/EMERG                   0
TEMP_ULTIMA/LAST_URG/EMERG                    0
GLU_ULTIMA/LAST_URG/EMERG                     0
SAT_02_ULTIMA/LAST_URG/EMERG                  0
TA_MAX_ULTIMA/LAST_URGEMERG                   0
TA_MIN_ULTIMA/LAST_URG/EMERG                  0
dtype: int64
```

Il·lustració 37. Nombre de valors nuls per a cada columna de la Taula 1

S'han detectat 12 columnes amb valors nuls, i s'han analitzat per ordre:

- Les columnes que indiquen les **dates** d'entrada i sortida a l'**UCI**, tenen un total de **2106 valors nuls** de 2307. Com que son valors que no es poden simular i no aporten gaire al model, s'ha decidit eliminar-les.
- La columna que indica els **dies a la UCI** té **2088 valors nuls**. Això vol dir que 2088 pacients no han requerit ser ingressats a l'UCI, i per tant, s'han emplenat els buits amb "0" (dies).
- La columna que indica l'**hora** de les primeres constants preses té **457 valors nuls**. Igual que l'anterior, com que és una variable que no es pot simular, s'ha decidit eliminar-la.
- La columna que indica la **raó d'alta** té **259 valors nuls**. En aquest cas, s'ha decidit emplenar-los amb "NO ALTA", ja que corresponen a pacients que encara no han sigut donats d'alta (veure Il·lustració 38)
- La columna que indica la **data d'alta** té **218 valors nuls**. Un altre cop, al tractar-se d'una data que no es pot simular, s'ha decidit eliminar-la.
- Les columnes que indiquen la **data** i **hora** d'admissió a urgències i l'**hora** de les últimes constants preses, tenen **83 valors nuls**. Igual que la resta de dates i hores, s'ha decidit eliminar-les.
- Les columnes que indiquen el **departament**, la **diagnosi** i el **destí** d'urgències, tenen també **83 valors nuls**. En aquest cas, s'han considerat columnes importants, i per això s'han mantingut, i s'han eliminat només les files amb valors nuls.

ADMISSION_D	_ENTRADA_UC/ICU_DATE_IN	ALIDA_UCI/ICU_DATE_	_DIAS/ICU_D	F_ALTA/DISCHARGE_DATE_ING	MOTIVO_ALTA/DESTINY_DISCHARGE_ING
11/03/2020	15/03/2020 13:41	12/04/2020 16:37	28	nan	nan

Il·lustració 38. Fila de la Taula 1 on es veu que un pacient ingressat que no ha sigut donat d'alta, no té el valor "MOTIVO_ALTA"

Amb aquests canvis, ja tenim el conjunt de dades net de valors buits.

Pas 3: Definir variables de sortida

Les variables de sortida seran úniques i diferents per a cada model:

- **Model 1:** estat futur del pacient, equivalent a la columna "MOTIVO_ALTA".
- **Model 2**: dies d'ingrés a la UCI, equivalent a la columna "UCI_DIAS".

En el cas de l'UCI, no hem de fer cap canvi, però, el model 1 té una peculiaritat: el recompte d'algunes classes és molt petit (veure Il·lustració 39). A més, ha aparegut una nova classe: "NO ALTA", que suma un total de sis classes. Per aquesta raó, la classificació es farà només per les tres primeres classes amb més valors, ja que la resta no arriben ni a les 100 files.

"Domicilio", "Fallecimiento" i "NO ALTA" es mantenen com a classes, mentre que les files amb "Traslado al Hospital", "Traslado a un Centro Sociosanitario" i "Alta Voluntaria" s'eliminen.

Domicilio	1577
Fallecimiento	313
NO ALTA	240
Traslado al Hospital	59
Traslado a un Centro Sociosanitario	32
Alta Voluntaria	3

Possibles classificacions

Il·lustració 39. Recompte de les classes dins de "MOTIVO_ALTA"

Pas 4: Codificar valors no numèrics

L'últim pas de l'anàlisi i normalització consisteix a codificar qualsevol valor que no sigui numèric, sinó categòrics. La raó de fer-ho és que els models de *Machine Learning* actuen millor quan han de treballar amb nombres, ja que poden decidir més fàcilment com categoritzar-los.

Aquesta codificació s'ha fet per les columnes amb cadenes de caràcters (veure Il·lustració 40) amb la funció "LabelEncoder" (veure Il·lustració 41) que proporciona la llibreria Scikit-Learn.

```
df_clean['sex'] = LabelEncoder().fit_transform(df_clean['sex'])
df_clean['sex'].value_counts()
```

Il·lustració 40. Codificació de la columna "Sexe" de la Taula 1

```
from sklearn.preprocessing import LabelEncoder
```

Il·lustració 41. Importació del mètode "LabelEncoder"

En aquest punt les dades ja estan llestes per utilitzar-se.

Taules 2 i 3

Per les taules 2 i 3 es va seguir el mateix procés de normalització, però finalment no es van utilitzar pels models.

5.1.2 Models

Tots dos models del projecte partiran del mateix conjunt de dades, el *DataFrame* de la Taula 1 de la base de dades. Però, al fer classificacions diferents, els entrenaments també seran diferents.

A continuació, s'explicarà el procés d'entrenament de cadascun d'ells.

5.1.2.1 Model 1: Predicció de l'estat dels pacients

El model 1 serà un **classificador** no pas un de regressió, això vol dir que el seu resultat serà un d'entre les classes: "**Domicilio**", "**Fallecimiento**" o "**NO ALTA**", i no un valor numèric.

S'han fet servir diferents mètodes i algorismes (*Random Forests*, SVM i KNN) amb l'objectiu de d'obtenir el model amb millor *accuracy*.

També s'ha d'aclarir que s'observaran dos mesures, i cal tenir en compte que no son les mateixes:

- La manera habitual de conèixer l'*accuracy* es testejar el model entrenat sobre un tros del conjunt de dades, el ***Test Set***. Podria simular els resultats que obtindria el model amb **valors nous**.
- El ***Cross-Validation***, permet comprovar l'*accuracy* del model dins d'un propi conjunt de dades, per evitar desviacions els resultats causats per grups no balancejats. Aquest mètode **no obté dades externes**.

Definir X, Y, *Training Set* i *Test Set*

Abans de començar l'entrenament en si, cal especificar quines dades del conjunt pertanyen a les variables d'entrada (etiquetades com "X") i quina la de sortida ("Y"). En aquest cas:

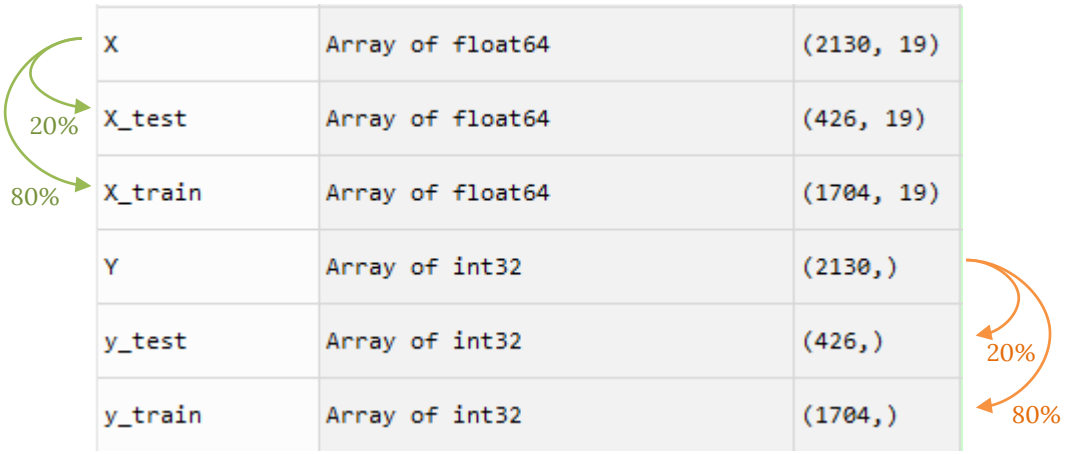
X = tot el conjunt, excepte la columna "MOTIVO_ALTA"
Y = la columna "MOTIVO_ALTA" (a predir)

Seguidament, sobre "X" s'ha tornat a fer una normalització, per a que totes les dades es trobin a la mateixa escala (veure Il·lustració 42), d'aquesta manera, s'evita l'existència de variables amb valors extrems que podrien alterar el model.

	3	4	5	6	7
0	0.27027	0.75	0.326923	1	0
1	0	0.75	0.153846	1	0.925187
2	0	0.625	0.230769	1	0.897756
3	0	0.75	0.153846	1	0.942643
4	0	0.75	0.115385	1	0.895262
5	0.0540541	0.75	0.692308	1	0.887781
6	0	0.75	0.115385	1	0
7	0	0.75	0.153846	1	0

Il·lustració 42. "X" amb les dades a escala, entre 0 i 1

I sobre aquestes variables és on es fa la separació entre el *Train* i el *Test Set*, que s'ha decidit dividir-les en **80% pel *training*** i **20% pel *testing*** (veure Il·lustració 43).



Il·lustració 43. Repartiment del Train i Test Set

Cal destacar que el *Test Set* (tant X com Y) **romandrà intacte** durant tot el procés, i les modificacions es faran només sobre el *Train Set*.

Prototipus 1: *Random Forest* amb dades inicials

Resum

Algorisme: *Random Forest*	**Testeig sobre *Test Set***: Si
Tipus de dades: Inicials, intactes	***Cross-Validation*:** No

Aquest és el primer entrenament que s'ha fet amb el model amb les dades inicials, no s'ha fet cap modificació excepte la separació en *Train* i *Test*.

S'ha emprat l'algorisme *Random Forest* (veure Il·lustració 44) amb els paràmetres per defecte de la funció establerts per la llibreria Scikit-Learn. Aquesta utilitza 100 arbres de decisió, tot i així, el nombre es pot especificar amb el paràmetre *n_estimators* si es desitja un altre valor.

```
#Use RANDOM FOREST CLASSIFIER to make predictions
model = RandomForestClassifier(random_state = 32)
# Train the model on training data
model.fit(X_train, y_train)
prediction = model.predict(X_test)
```

Il·lustració 44. Entrenament del primer prototipus amb RF[3]

El model s'ha entrenat amb les variables d'entrada (X_train) i sortida (y_train) del conjunt de *Training*, i la predicció s'ha fet sobre el *Test Set* (X_test) reservat a l'inici.

Com s'ha mencionat a la introducció, s'han fet servir **tres mètriques** diferents per a comprovar la correctesa del model, recordem què representen cadascuna d'elles:

- L'***accuracy*** és la proporció entre valors encertats i el nombre total de valors. Es calcula amb tots els valors de la matriu.
- La **precisió** és la proporció entre els valors positius encertats i tots els valors classificats com a pertanyents a la classe (positius falsos i veritables). Es calcula com a la mitjana de totes les classes.
- El ***Threat Score***, o **IoU**, és la proporció entre els valors positius encertats i la suma d'aquesta més tots els desencerts. Es calcula com a la mitjana de totes les classes.

Aquests valors es calculen mitjançant la **matriu de confusió**, que s'ha generat amb el mètode "*confusion_matrix()*" (veure Il·lustració 45 i 46), que obté el valor predit i el valor real.

```
from sklearn.metrics import confusion_matrix, plot_confusion_matrix, recall_score
```

Il·lustració 45. Importació del mètode "confusion_matrix" i altres de la llibreria Scikit-Learn

[3] Al cridar el mètode "RandomForestClassifier", el paràmetre "*random_state*" es fa servir per fixar els valors obtinguts i no obtenir-ne d'aleatoris cada vegada que s'executa el codi.

```
#Confusion matrix
cm= confusion_matrix(y_test, prediction)
print(cm)
```

Il·lustració 46. Generació de la matriu de confusió

Aquests son els valors de la matriu resultant:

Classe real				
	Domicilio	296	13	3
	Fallecimiento	41	23	5
	NO ALTA	36	3	6
		Domicilio	*Fallecimiento*	NO ALTA
		Classe predita		

A primera vista ja es veu com hi ha una gran diferència d'encerts entre les classes, amb "Domicili" sent la més exitosa. Comprovem ara els resultats de les mètriques:

Accuracy	$\frac{TP+TN}{P+N} = \frac{296+23+6}{426}$	**76'29%**
Precisió	**(79'36+58'97+42'86)/3**	**60,39%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{296}{296+41+36}$	79'36%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{23}{23+13+3}$	58'97%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{6}{6+3+5}$	42.86%
Threat Score/IoU	**(76'09+27'05+11'32)/3**	**38'16%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{296}{296+13+3+41+36}$	76'09%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{23}{23+41+5+13+3}$	27'05%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{6}{6+36+3+3+5}$	11'32%

Els resultats d'aquesta taula son la raó per la qual s'ha decidit comparar diferents mètriques de mesura. Si s'hagués calculat només l'*accuracy*, hauria semblat un model força competitiu, ja que compta els encerts sense tenir en compte les classes individuals.

En canvi, si es desglossa en classes, la realitat és que la capacitat de classificació ha disminuït, sobretot amb la fórmula del *Threat Score*, la més crítica degut a que inclou també els falsos negatius.

Però, a que es deu aquest desequilibri en les prediccions? Si es recorden el nombre de files per a cada classe de sortida, tenim (del conjunt sencer):

- "**Domicili**" = 1577 registres
- "**Fallecimiento**" = 313 registres
- "**NO ALTA**" = 240 registres

Clarament, aquestes dades no estan balancejades, causant que el model gairebé no reconegui a les minories. Davant d'aquest problema, es va trobar una primera solució: generar noves dades a partir de les existents, i així apareix un **segon prototipus**.

Prototipus 2: *Random Forest* i *Resample*

Resum

Algorisme: *Random Forest*	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades creant còpies	***Cross-Validation*:** No

Aquest prototipus utilitza el mateix algorisme que l'anterior, el *Random Forest*, el que canvia son les variables d'entrada, que s'han equilibrat amb l'objectiu de tenir el mateix nombre de dades de cada classe i esbrinar si el model millora.

El mètode que s'ha utilitzat es coneix com a ***Resampling*** (veure Il·lustració 47), permet triar el nombre de files que es vol per a cada possible classificació, i s'ha treballat només sobre les dades d'entrenament.

```
from sklearn.utils import resample
```

Il·lustració 47. Importació del mètode "Resample"

El funcionament és diferent en base al nombre de dades triades, si una classe té més files de les especificades, n'**elimina** la resta de **forma aleatòria**; en el cas de contenir un nombre inferior, agafa files aleatòriament i va generant **còpies** fins a arribar al número demanat.

En aquest cas, s'ha decidit que aquest nombre serà **1000**, és a dir, 1000 files per a cada classe (veure Il·lustració 48), que suma un total de 3000 files al *Train Set*.

```
df1 = resample(dataset1,
                                replace=True,     # sample with replacement
                                n_samples=1000,    # to match average class
                                random_state=32)

df2 = resample(dataset2,
                                replace=True,     # sample with replacement
                                n_samples=1000,    # to match average class
                                random_state=32)

df3 = resample(dataset3,
                                replace=True,     # sample with replacement
                                n_samples=1000,    # to match average class
                                random_state=32)
```

Il·lustració 48. Crida al mètode "resample" per generar noves dades

Per a cridar aquest mètode, abans s'ha hagut de separar cada classe en diferents blocs per a fer les còpies i al finalitzar el procés s'han tornat a unir en el mateix *Data Set*, que ha servit pel *training*.

L'entrenament de l'algorisme ha sigut igual que el prototipus 1, i amb els mateixos paràmetres per defecte (veure Il·lustració 49).

```
RF_model_resampled = RandomForestClassifier(random_state = 32)
```

Il·lustració 49. Entrenament del model amb les dades regenerades

La manera de comprovar els resultats es manté igual, s'ha obtingut primer doncs, la matriu de confusió:

Classe real		Classe predita		
		Domicilio	*Fallecimiento*	NO ALTA
	Domicilio	266	31	15
	Fallecimiento	28	34	7
	NO ALTA	23	8	14

Igual que abans, sembla haver-hi una gran diferència entre la classe "Domicilio" i la resta. A continuació es mostren els resultats de les mètriques:

Accuracy	$\frac{TP+TN}{P+N} = \frac{266+34+14}{426}$	**73'71%**
Precisió	**(83'91+46'58+38'89)/3**	**56'46%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{266}{266+28+23}$	83'91%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{34}{34+31+8}$	46'58%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{14}{14+15+7}$	38'89%
Threat Score/IoU	**(73'28+31'48+20'89)/3**	**41,89%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{266}{266+31+15+28+23}$	73'28%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{34}{34+28+7+31+8}$	31'48%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{14}{14+23+8+15+7}$	20'89%

S'observa que els resultats han millorat una mica, sobretot pels casos específics de les classes, tot i així, segueix el desequilibri amb "Fallecimiento" i "NO ALTA", les classes amb menys dades originals.

Això demostra que el *resampling* de les dades **no ha tingut un gran impacte** sobre el model, perquè els "nous" valors generats ja eren coneguts al tractar-se de còpies pures. Per això, el següent prototipus proposa un altre mètode per a regenerar dades.

Prototipus 3: *Random Forest* i SMOTE

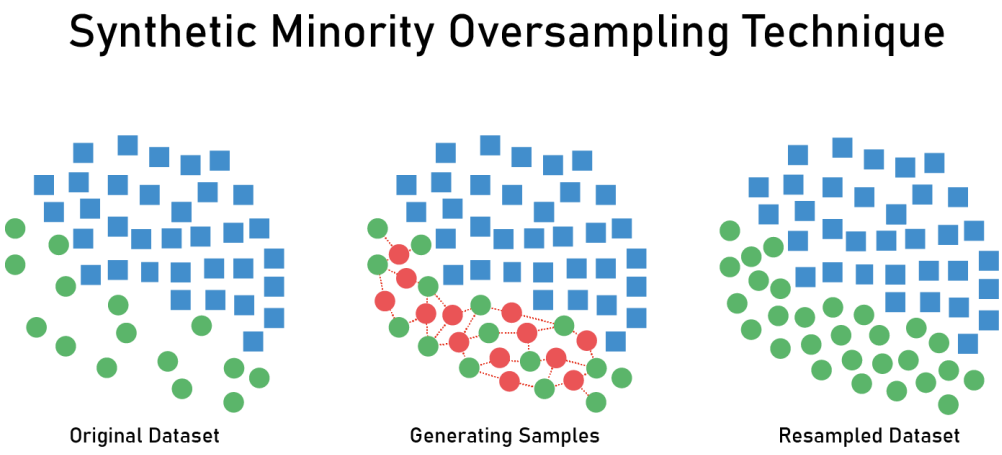
Resum

Algorisme: *Random Forest*	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades creant nous valors	***Cross-Validation*:** Si

En aquest prototipus és on es fa servir la llibreria **Imblearn**, per a generar noves dades, amb un procés anomenat ***oversampling***.

La diferència principal amb l'anterior és que les noves variables no son còpies de les existents i en comptes de proporcionar un nombre al qual adaptar la quantitat de files, es fa un augment de les classes minoritàries.

Per simular dades, el que fa el mètode es trobar valors límits de les classes minoritàries (veure Il·lustració 50), i procedeix a crear de manera aleatòria informació dins d'aquests rangs.



Il·lustració 50. Funcionament del SMOTE. Tècnica de generació de noves dades
Analytics Vidhya, "Bank Data: SMOTE", 2020

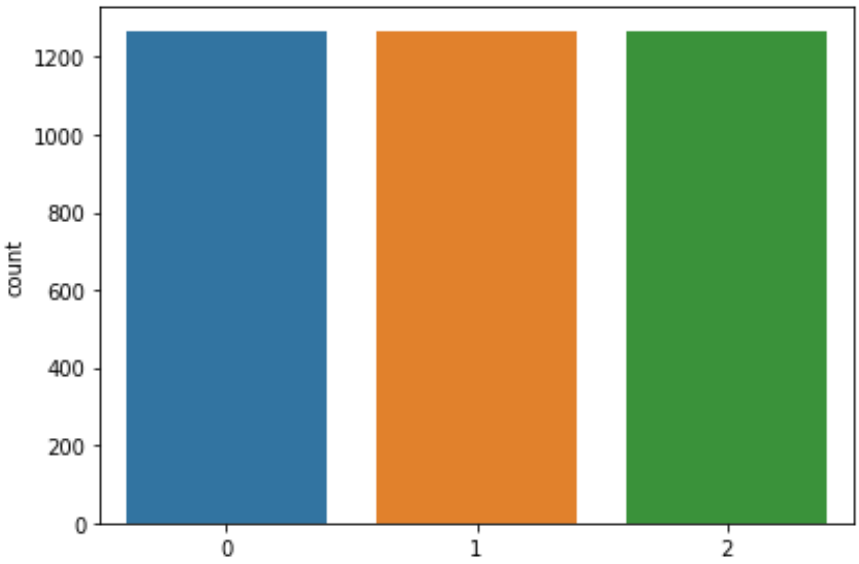
A la imatge es veu clarament com s'identifiquen les diferents classes, i dins dels límits de la classe petita (la verda), va afegint nous valors per a equilibrar-les.

La funció en concret, s'anomena "**SMOTE()**", i s'ha fet servir just abans d'entrenar l'algorisme (RF) amb el nou conjunt d'enterament proporcionat per ella (veure Il·lustració 51).

```
from imblearn.over_sampling import SMOTE
oversampler = SMOTE(random_state=32)
X_over, y_over = oversampler.fit_resample(X_train, y_train)
RF_model_over = RandomForestClassifier(random_state = 32)
```

Il·lustració 51. Crida al mètode "SMOTE" per obtenir les noves dades d'entrenament

Les variables "X_over" i "y_over", que han quedat balncejades (veure Il·lustració 52), representen el *Train Set* amb dades simulades, i a partir d'aquí, es faran servir les mateixes (queden fixes) quan es requereixi fer *oversampling*.



Il·lustració 52. Histograma de "y_over" amb noves dades generades
D'esquerra a dreta: "Domicilio", "Fallecimiento", "NO ALTA"

Un cop entrenat el model i demanada una predicció, seguint els mateix passos, s'obté la matriu de confusió:

Classe real		Classe predita		
		Domicilio	*Fallecimiento*	NO ALTA
	Domicilio	256	37	19
	Fallecimiento	21	36	12
	NO ALTA	22	9	14

De la qual s'obtenen els següents resultats de les mètriques:

Accuracy	$\frac{TP+TN}{P+N} = \frac{256+36+14}{426}$	**71'83%**
Precisió	**(85'61+43'90+31'11)/3**	**53'54%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{256}{256+21+22}$	85'61%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{36}{36+37+9}$	43'90%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{14}{14+19+12}$	31'11%
Threat Score/IoU	**(72'11+31'30+18'42)/3**	**40'61%**

"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{256}{256+37+19+21+22}$	72'11%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{36}{36+21+12+37+9}$	31,30%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{14}{14+22+9+19+12}$	18,42%

Els **resultats obtinguts son menors** a aquells obtinguts amb el prototipus 2, en totes les mètriques: l'*accuracy*, la precisió i el *Threat score*. Un altre cop, es deu a la poca varietat de les classes "Fallecimiento " i "NO ALTA", que no proporcionen un marge suficientment ampli per generar variables valioses.

Cross-Validation

En apartats anteriors s'ha explicat el mètode del *Cross-Validation* (veure 2.1.1.1 Aprenentatge Supervisat), i en aquest prototipus s'ha empleat per a descartar que els resultats no s'han vist afectats per variables mal repartides entre el *Train* i el *Test Set*.

El conjunt de dades sobre el qual s'ha portat a terme han sigut dos:

1. El conjunt de dades d'aprenentatge augmentat (*Train Set Oversampled*): "X_over" i "y_over".
2. El conjunt anterior més el *Test Set* inicial intacte: "X_test" i "y_test".

Aquests conjunts s'han dividit en **10 sub-grups** (*K Folds*), dels quals 9 s'han utilitzat per entrenar un model i el restant per representar el grup de validació (el *Test Set* dins de l'entrenament).

De cada model entrenat, s'ha demanat una predicció i s'ha guardat el resultat en una matriu de confusió, que al final del procés, ha permès comprovar el comportament del model. Aquests han sigut els resultats obtinguts, on es pot veure que **hi ha més dades** per a cada classe, ja que s`ha validat tot el conjunt i no només el *Test Set*.

Train Set Oversampled

Classe real		*Domicilio*	*Fallecimiento*	NO ALTA
	Domicilio	1029	131	105
	Fallecimiento	74	1112	79
	NO ALTA	99	89	1077

Classe predita

A primera vista, sembla que els encerts estan més balancejats. Es comproven els resultats en detall amb les mètriques que s'han fet servir fins ara:

Accuracy	$\frac{TP+TN}{P+N} = \frac{1029+1112+1077}{3795}$	**84'79%**

Precisió	**(85'61+83'48+85'41)/3**	**84'83%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{1029}{1029+74+99}$	85'61%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{1112}{1112+131+89}$	83'48%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{1077}{1077+105+79}$	85'41%
Threat Score/IoU	**(72'11+31'30+18'42)/3**	**73'59%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{1029}{1029+131+105+74+99}$	71'56%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{1112}{1112+74+79+131+89}$	74,88%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{1077}{1077+99+89+105+79}$	74,33%

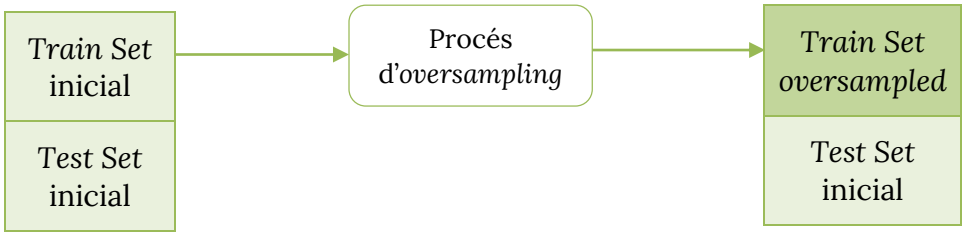
Es veu que els percentatges son **força alts** en comparació a les prediccions fetes sobre el *Train Set*. Si ens fixem en el valor de cada classes, fins i tot amb la mètrica més crítica (*Threat Score*), totes elles passen el 70% de prediccions correctes.

A què es pot deure aquest fet? La base d'aquests resultats, és que el model ja coneixia mínimament les dades que havia de predir.

Quan es va passar pel procés d'*oversampling*, el *Train Set* va augmentar en mida emplenant-se de valors "inspirats" dels reals, això vol dir que per les classes minoritàries, les dades no son variades, sinó **molt** similars. Aquest fet causa, que a l'hora de dividir el conjunt en *K Folds*, les variables al grup d'entrenament i les variables a predir siguin gairebé les mateixes, permetent al model reconèixer-les ràpidament.

***Train Set Oversampled* + Test Set**

Per a aquest conjunt, s'ha utilitzat l'anterior, afegit el *Train Set* inicial (veure Il·lustració 53), que no ha estat modificat, quedant d'aquesta manera:



Il·lustració 53. Agrupació de Train Set augmentat i el Test Set

Els passos per comprovar el seu funcionament, han sigut els mateixos, començant per la **matriu de confusió**:

Classe real		Classe predita		
		Domicilio	*Fallecimiento*	NO ALTA
	Domicilio	1324	147	106
	Fallecimiento	101	1146	87
	NO ALTA	145	107	1058

Si s'observen les prediccions positives de cada classe (*True Positives*) podria semblar que el nombre d'encerts és major, però això és perquè s'han afegit les dades del *Train Set*, augmentant els valors totals. Per això, seguidament es troben els resultats de les mètriques:

Accuracy	$\frac{TP+TN}{P+N} = \frac{1324+1146+1058}{4221}$	**83'58%**
Precisió	**(84'33+81'86+84'57)/3**	**83'59%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{1324}{1324+101+145}$	84'33%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{1146}{1146+147+107}$	81'86%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{1058}{1058+106+87}$	84'57%
Threat Score/IoU	**(72'11+31'30+18'42)/3**	**71'73%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{1324}{1324+147+106+101+145}$	72'63%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{1146}{1146+101+87+147+107}$	72,17%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{1058}{1058+145+107+106+87}$	70,39%

El percentatges aquest son bastant bons, però han baixat una mica respecte l'anterior *Cross-Validation*. La raó ara, es que al afegir les noves dades del *Train Set*, que no s'han fet servir per balancejar les classes, el model es troba amb alguns valors no gaire familiars.

Tot i així, es pot dir que tant els resultats d'aquesta validació, com la primera, han estat força positius.

Prototipus 4: *Random Forest* amb els millors paràmetres

Resum

Algorisme: *Random Forest* (millors paràmetres)	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades creant nous valors	***Cross-Validation*:** Si

Aquest prototipus s'ha creat amb l'objectiu de millorar el model utilitzant *Random Forests*, pels anteriors, s'han fet servir els paràmetres per defecte. Aquest cop, s'ha fet una iteració amb un nombre diferent d'arbres de decisió per trobar quin aconsegueix l'*accuracy* més elevada.

El nom del paràmetre que indica aquest valor és "***n_estimators***", iterant sobre: 25, 50, 100 i 200 arbres. S'ha generat un bucle per utilitzar els diferents nombres d'arbres, que al mateix cop, crida a una funció que entrena un model fent servir els *Random Forests* amb diferents estimadors (veure Il·lustració 54).

```
def score_model(params):
    global smote_cm
    RF_model_over_cross = RandomForestClassifier(**params)
    for j in range(len(X_train_folds)):
            # Fit the model on the upsampled training data
            model_obj = RF_model_over_cross.fit(X_train_folds["x"+ str(j)], y_train_folds["y" + str(j)])
            # Score the model on the (non-upsampled) validation data
            prediction =  model_obj.predict(X_val_folds["x"+ str(j)])
            score = recall_score(y_val_folds["y" + str(j)], prediction , average=None)
            scores.append(score)
            smote_cm = np.add(smote_cm, confusion_matrix(y_val_folds["y" + str(j)], prediction))
            return scores
    #Let's make a loop to find the best parameters for the model
score_tracker = []
for n_estimators in [25, 50, 100, 200]:
    example_params = {
        'n_estimators': n_estimators,
        'random_state': 32
    }
    example_params['recall'] = np.mean(score_model(example_params))
    score_tracker.append(example_params)
```

Il·lustració 54. Bucle que itera sobre diferents estimadors per trobar els millors paràmetres

El funcionament és similar al del *Cross-Validation*, s'ha dividit el *Train Set*, aquest cop en 3 *folds*, dels quals 2 s'han entrenat amb els diferents "*n_estimators*" i s'ha validat amb el *fold* restant. D'aquestes validacions, s'ha agafat el nombre d'arbres que millor resultat ha donat (veure Il·lustració 55), que ha sigut **50**.

```
Out[4]: {'n_estimators': 50, 'random_state': 32, 'recall': 0.5022190377299025}
```

Il·lustració 55. Nombre d'estimadors (arbres) que han generat els millors resultats

A partir d'aquí, s'han fet dues proves amb "*n_estimators* = 50" al *Random Forest*:

1. Amb l'*oversampling* original sobre tot el *Train Set*.
2. Dividint el *Train Set* en diversos *folds* i fent *oversampling* sobre cadascun d'ells, i a més, s'ha fet també una validació amb *Cross-Validation*.

Oversampling* del *Train Set

S'ha fet servir l'algorisme *Random Forest* amb 50 estimadors, s'ha fet la predicció sobre el *Test Set* original i s'ha obtingut la següent **matriu de confusió**:

Classe real		Classe predita		
	Domicilio	255	39	18
	Fallecimiento	22	35	12
	NO ALTA	20	11	14
		Domicilio	*Fallecimiento*	NO ALTA

Son uns resultats força similars al primer *oversampling*, i si s'observen les diferents mètriques, també es comprova que son gairebé els mateixos:

Accuracy	$\frac{TP+TN}{P+N} = \frac{255+35+14}{426}$	**71'36%**
Precisió	**(85'86+41'17+31'82)/3**	**52'95%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{255}{255+22+20}$	85'86%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{35}{35+39+11}$	41'17%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{14}{14+18+12}$	31'82%
Threat Score/IoU	**(72'03+29'42+18'67)/3**	**40'03%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{255}{255+39+18+22+20}$	72'03%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{35}{35+22+12+39+11}$	29,41%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{14}{14+20+11+18+12}$	18,67%

Oversampling* de cada *fold* del *Train Set

Aquest cop s'ha fet el procés d'*oversampling* sobre els 3 *folds* que s'han fet servir per trobar els millors paràmetres. Això vol dir que no s'ha agafat el *Train Set* sencer com a base per crear noves dades, sinó aquelles dins d'un *fold*.

La **matriu de confusió** resultant, un cop entrenat el model amb la combinació dels 3 *folds* i el millor paràmetre ("*n_estimators* = 50"), és la següent:

Classe real		Classe predita		
	Domicilio	244	38	30
	Fallecimiento	22	34	13
	NO ALTA	23	9	13
		Domicilio	*Fallecimiento*	NO ALTA

Sembla que el nombre d'errors han augmentat una mica (es pot veure amb la predicció [38] de "Fallecimiento" o la predicció de "NO ALTA" [30], que realment eren "Domicilio"). Si calculem totes les mètriques obtenim:

Accuracy	$\frac{TP+TN}{P+N} = \frac{244+34+13}{426}$	**68'31%**
Precisió	**(84'43+41'96+23'21)/3**	**49'87%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{244}{244+22+23}$	84'43%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{34}{34+38+9}$	41'96%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{13}{13+30+13}$	23,21%
Threat Score/IoU	**(68'35+29'31+14'77)/3**	**37'48%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{244}{244+38+30+22+23}$	68'35%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{34}{34+22+13+38+9}$	29,31%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{13}{13+23+9+30+13}$	14,77%

Els percentatges son encara inferiors a l'*oversampling* pel *Train Set* complert, això demostra que cada *fold* no contenia suficient informació per a generar dades valuoses, la qual cosa ha fet que el model no disposés de valors nous dels quals aprendre.

Cross-Validation

A mode de prova, s'ha tornat a fer el *Cross-Validation* (dividint en 5 grups) amb un conjunt de dades format per *folds* que han passat el procés d'*oversampling*, amb l'objectiu de trobar si era preferible separar el *Train Set* en grups diferents per entrenar un model.

Com anteriorment, s'ha generat la **matriu de confusió**, amb els següents valors. S'ha de aclarir que ara el total de valors és diferent al conjunt inicial, ja que s'han afegit dades segons la diferència de classes de cada *fold*:

Classe real		Classe predita		
	Domicilio	829	108	63
	Fallecimiento	79	86	31
	NO ALTA	95	46	27
		Domicilio	*Fallecimiento*	NO ALTA

A diferència de les altres validacions, aquesta sembla no tenir bons resultats amb les classes minoritàries, el mateix comportament que s'obté al fer la predicció sobre el *Test Set*.

Els percentatges obtinguts a partir de les diferents mètriques son:

Accuracy	$\frac{TP+TN}{P+N} = \frac{829+86+27}{1364}$	**69'06%**
Precisió	**(82'65+35'83+22'32)/3**	**46'93%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{829}{829+79+95}$	82'65%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{86}{86+108+46}$	35'83%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{27}{27+63+31}$	22'32%
Threat Score/IoU	**(72'11+31'30+18'42)/3**	**35'16%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{829}{829+108+63+79+95}$	70'61%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{86}{86+79+31+108+46}$	24'57%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{27}{27+95+46+63+31}$	10,31%

Com mostra la matriu de confusió, els encerts per les classes petites tornen a ser inferiors.

Els resultats d'aquest prototipus, la versió amb l'*oversampling* per a cada *fold*, ha donat els percentatges d'encert més baixos fins ara, tant a la predicció sobre el *Test Set* com amb la validació.

Prototipus 5: *Support Vector Machines* (SVM)

Resum

Algorisme: *Support Vector Machines* (SVM)	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades creant nous valors	***Cross-Validation:*** No

En aquest prototipus es fa servir el segon algorisme mencionat, el *Support Vector Machines* (SVM), amb l'objectiu de competir amb els *Random Forest* i comprovar si un altre mètode millora les prediccions, o confirmar que la dificultat de trobar un bon model resideix en les classes no balancejades.

El funcionament és el mateix als anteriors prototipus, es crida la funció de l'algorisme i s'especifiquen els paràmetres si es veu necessari (en aquest cas s'han utilitzat els que estan per defecte) i a continuació s'entrena el model (veure Il·lustració 56) amb les variables d'entrada.

```
from sklearn import svm

SVM_oversampled = svm.LinearSVC()
SVM_oversampled.fit(X_over, y_over)

#Predict
svm_prediction = SVM_oversampled.predict(X_test)

#Accuracy
print("Accuracy SVM = ", metrics.accuracy_score(y_test, svm_prediction))
```

Il·lustració 56. Crida i entrenament amb l'algorisme SVM

I per comprovar els resultats, també es genera la **matriu de confusió**:

Classe real		Classe predita		
		Domicilio	*Fallecimiento*	NO ALTA
	Domicilio	222	54	36
	Fallecimiento	4	51	14
	NO ALTA	12	17	16

A partir d'aquesta, s'han calculat les següents mètriques:

Accuracy	$\frac{TP+TN}{P+N} = \frac{222+51+16}{426}$	**67'84%**
Precisió	**(93'28+41'80+24'24)/3**	**53'11%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{222}{222+4+12}$	93'28%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{51}{51+54+17}$	41'80%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{16}{16+36+14}$	24,24%
Threat Score/IoU	**(67'68+36'43+16'84)/3**	**40'32%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{222}{222+54+36+4+12}$	67'68%

"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{51}{51+4+14+54+17}$	36,43%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{16}{16+12+17+36+14}$	16,84%

L'única millora que es pot observar respecte els *Random Forests* és les prediccions de la classe "Domicilio". La resta, l'*accuracy*, la precisió i el *Threat Score* han tingut valors inclús inferiors, derivats de la quantitat d'errors a l'hora de classificar classes minoritàries, sobre tot "NO ALTA".

Prototipus 5: *K-nearest-neighbors* (KNN)

Resum

Algorisme: K-*nearest neighbors* (KNN)	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades creant nous valors	***Cross-Validation*:** No

L'últim algorisme que s'ha empleat és el K-*nearest-neighbors* (KNN), que es fa servi sovint quan es disposa d'un nombre limitat de dades per entrenar. S'ha fet servir per afegir un nou mètode i comprovar quin dona millor resultats.

La manera d'implementar-lo és la mateixa als anteriors: es crida la funció de l'algorisme amb els paràmetres necessaris, i s'entrena el model sobre el conjunt de dades d'entrada (veure Il·lustració 57).

```
from sklearn.neighbors import KNeighborsClassifier

knn_oversampled = KNeighborsClassifier()

knn_oversampled.fit(X_over, y_over)
```

Il·lustració 57. Crida i entrenament amb l'algorisme KNN

Després d'entrenar el model, el següent pas ha sigut generar la **matriu de confusió**, com s'ha fet fins ara, per veure els encerts i els errors:

Classe real				
	Domicilio	215	49	48
	Fallecimiento	17	35	17
	NO ALTA	16	10	19
		Domicilio	*Fallecimiento*	NO ALTA
		Classe predita		

S'observa que els encerts per a cada classe han disminuït una mica, i es manté la diferència entre classes majoritàries i minoritàries.

A través de la matriu, s'han calculat les mètriques han obtingut aquests resultats:

Accuracy	$\frac{TP+TN}{P+N} = \frac{215+35+19}{426}$	**63'15%**
Precisió	**(86'69+37'23+22'62)/3**	**48'85%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{215}{215+17+16}$	86'69%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{35}{35+49+10}$	37'23%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{19}{19+48+17}$	22'62%
Threat Score/IoU	**(62'32+27'34+17'27)/3**	**35'64%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{215}{215+49+48+17+16}$	62'32%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{35}{35+17+17+49+10}$	27,34%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{19}{19+16+10+48+17}$	17,27%

Aquest cop les mètriques han obtingut percentatges encara menor que amb l'algorisme SVM, sobretot quan s'observa el desglossament per classes.

Fins a aquest punt, s'han fet proves amb tres algorismes diferents, amb *Random Forests* sent amb el que millors prediccions s'han obtingut. Per aquesta raó, s'han fet **3 proves més** utilitzant aquest algorisme, amb el mateix objectiu: **millorar l'*accuracy***.

Prototipus 6: *Random Forest + Undersampling*

Resum

Algorisme: *Random Forests*	**Testeig sobre *Test Set***: Si
Tipus de dades: Equilibrades eliminant valors	***Cross-Validation*:** No

En els anteriors prototipus s'havia fet un tipus de modificació sobre les dades inicial per equilibrar les classes, en aquest, s'ha utilitzat el mètode contrari: eliminar dades de les classes majoritàries per balancejar-les.

S'ha emprat la llibreria **Imblearn**, que proporciona el mètode "*RandomUnderSampler()*" per a disminuir les dades de forma aleatòria. De la mateixa manera que abans, es crida la funció s'especifiquen els paràmetres si es necessari. Un cop fet, s'entrena el model i es demana la predicció sobre el *Test Set* (veure Il·lustració 58).

```
from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler(random_state=0)
X_under, y_under = rus.fit_resample(X_train, y_train)
RF_under = RandomForestClassifier(random_state=32)

RF_under.fit(X_under, y_under)
RF_under_prediction = RF_under.predict(X_test)
```

Il·lustració 58. Crida i entrenament amb les dades disminuïdes amb "RandomUndersampler"

Al fer el procés d'*undersampling*, evidentment, el nombre de dades per entrenar disminueix, resultant en **585 files**, però el *Test Set* segueix sent el mateix, sense cap modificació, a partir del qual s'obté la **matriu de confusió**:

Classe real		Classe predita		
		Domicilio	*Fallecimiento*	NO ALTA
	Domicilio	190	51	71
	Fallecimiento	5	43	21
	NO ALTA	12	12	21

Al calcular les mateixes mètriques que s'han fet servir fins ara, els resultats son:

Accuracy	$\frac{TP+TN}{P+N} = \frac{190+43+21}{426}$	**59'62%**
Precisió	**(91'79+40'57+18'58)/3**	**50'31%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{190}{190+5+12}$	91'79%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{43}{43+51+12}$	40'57%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{21}{21+71+21}$	18'58%
Threat Score/IoU	**(57'75+32'58+15'33)/3**	**35'22%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{190}{190+51+71+5+12}$	57'75%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{43}{43+5+21+51+12}$	32'58%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{21}{21+12+12+71+21}$	15'33%

Un cop més, hi ha una gran diferencia entre els encerts de les diferents classes. Ara que s'han disminuït els valors de la classe "Domicili", segueix sent la que ha aconseguit la precisió més alta.

Prototipus 7: *Random Forest* + Eliminació d'una classe

Resum

Algorisme: *Random Forests*	**Testeig sobre Test Set**: Si
Tipus de dades: Equilibrades creant nous valors	***Cross-Validation:*** No

Els resultats obtinguts en els prototipus anteriors tenen una cosa en comú: el menor nombre d'encerts son sempre els de la classe "NO ALTA". S'ha pensat que una classificació com aquesta, no seria de gran ajuda per als metges, ja que només indica que seguiran ingressats.

Per això, en aquest prototipus s'ha fet una prova **eliminant** aquesta classe per complert, i fent una classificació entre les altres dues: "Domicilio" i "Fallecimiento", resultant en un total de **1890 registres** que han passat pel procés d'*oversampling*.

```
## Drop rows where Discharge reason is "No Alta" or a transfer
df_two_binary = df_two_binary.drop(df_two_binary[df_two_binary["discharge_reason"] == "NO ALTA"].index)
```

Il·lustració 59. Eliminació de la classe "NO ALTA"

A partir d'aquí, els passos a seguir son els mateixos, cridar a l'algorisme *Random Forests*, entrenar-lo i demanar una predicció, de la qual s'ha generat la **matriu de confusió**.

Classe real		*Domicilio*	*Fallecimiento*
	Domicilio	275	36
	Fallecimiento	29	38

Classe predita

Es torna a veure com el comportament és millor per a una classe, la "Domicilio". Però si observem les mètriques, es veu que les mitjanes si han millorat:

Accuracy	$\frac{TP+TN}{P+N} = \frac{275+38}{378}$	**82'80%**
Precisió	**(90'46+51'35)/2**	**70'91%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{275}{275+29}$	90'46%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{38}{38+36}$	51'35%
Threat Score/IoU	**(80'88+36'89)/2**	**58'89%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{275}{275+36+29}$	80'88%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{38}{38+29+36}$	36'89%

El problema de classes no balancejades persisteix, tot i així, és el **resultat més alt** que s'ha obtingut fins ara de l'*accuracy* (82'80%).

Prototipus 7: *Random Forest* + Característiques més importants

Resum

Algorisme: *Random Forests*	**Testeig sobre *Test Set***: Si
Tipus de dades: Inicials (Top 10 d'importància)	***Cross-Validation*:** No

Finalment, un últim prototipus, pel qual també s'ha fet servir *Random Forests*, s'ha entrenat només amb les **10 primeres variables** que més aporten al model (veure Il·lustració 60).

age	0.214246
em_last_OxSaturation	0.082652
em_last_hr	0.074135
em_first_OxSaturation	0.072431
em_first_hr	0.069017
em_last_max_blood_pressure	0.062730
em_last_temperature	0.058450
em_first_temperature	0.057862
em_first_max_blood_pressure	0.055361
em_last_min_blood_pressure	0.053931
em_first_min_blood_pressure	0.053399
emergency_diagnosis	0.051235
icu_days	0.043519
sex	0.022904
department	0.012416
covid_diagnosis	0.008863
em_last_glucemia	0.003659
em_first_glucemia	0.003187
emergency_dept	0.000003

dtype: float64

TOP 10

Il·lustració 10. Importància de les variables durant l'entrenament del model

Es pot veure com l'edat del pacient és la variable que més aporta amb diferència, un **21'42%,** mentre que la resta ni tan sols arriben al 10%. Els 10 primers valors inclouen les dades preses a urgències, com: la saturació d'oxigen, pressió arterial, temperatura i el ritme cardíac.

A partir d'aquestes variables, s'ha creat un nou conjunt de dades, del qual s'han tornat a separar les entrades (X) i sortides (Y). Els passos següents son els mateixos que fins ara: cridar a l'algorisme, entrenar el model i fer una predicció.

Abans de mostrar els resultats, cal recordar que no s'ha passat per cap procés d'augment o disminució de dades, excepte l'eliminació de variables fora del rang.

La **matriu de confusió** resultant és aquesta:

Classe real		Classe predita		
	Domicilio	292	19	1
	Fallecimiento	42	25	2
	NO ALTA	37	5	3
		Domicilio	*Fallecimiento*	NO ALTA

Tornem a veure encerts no balancejats, i la classe que més confusió porta al model sembla ser "NO ALTA", que ha categoritzat com a "Domicilio" 37 cops. Amb aquesta matriu, el càlcul de les mètriques és:

Accuracy	$\frac{TP+TN}{P+N} = \frac{292+25+3}{426}$	**75'12%**
Precisió	**(86'65+51'02+50)/3**	**62'56%**
"Domicili"	$\frac{TP}{TP+FP} = \frac{292}{292+42+37}$	86'65%
"Fallecimiento"	$\frac{TP}{TP+FP} = \frac{25}{25+19+5}$	51'02%
"NO ALTA"	$\frac{TP}{TP+FP} = \frac{3}{3+1+2}$	50%
Threat Score/IoU	**(74'68+26'88+6'25)/3**	**35'94%**
"Domicili"	$\frac{TP}{TP+FN+FP} = \frac{292}{292+19+1+42+37}$	74'68%
"Fallecimiento"	$\frac{TP}{TP+FN+FP} = \frac{25}{25+42+2+19+5}$	26'88%
"NO ALTA"	$\frac{TP}{TP+FN+FP} = \frac{3}{3+37+5+1+2}$	6'25%

Aquests resultats demostren que les variables de les quals s'han prescindit no eren redundants, sinó que aportaven un gran valor, sobretot, a l'hora d'identificar les classes minoritàries, ja que es veu com les prediccions sobre la classe "NO ALTA" en caigut força.

5.1.2.1 Model 2: Predicció de dies a la UCI

El segon model també es un classificador, aquest cop per a predir entre un grups de dies, quants en podria necessitar un pacient l'UCI.

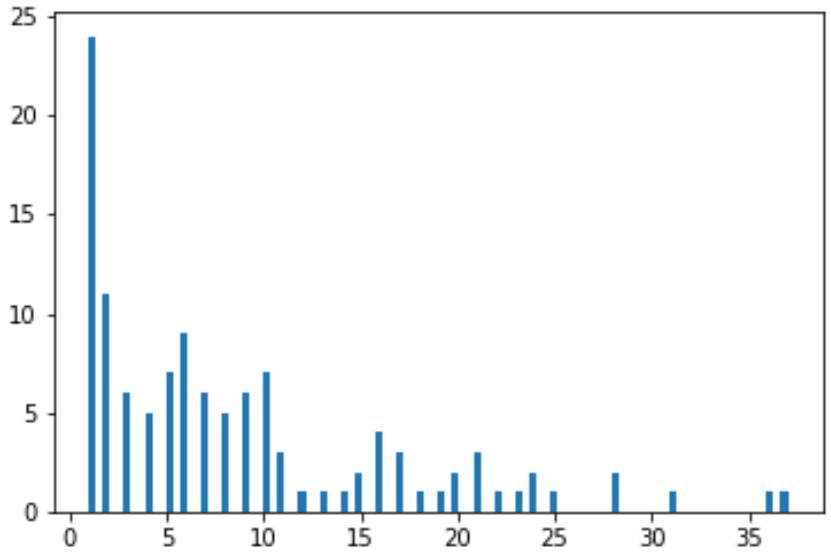
El procés de neteja de dades és exactament el mateix a l'anterior, la diferència arriba al moment de separar-les en variables d'entrada i sortida. En el model 1, les dades de sortida representaven la raó d'alta, en aquest en canvi, aquestes variables son:

X = tot el conjunt, excepte la columna "UCI_DIAS"
Y = la columna "UCI_DIAS" (a predir)

Aquesta columna, conté majoritàriament el valor "0", que pertany a pacients que no han sigut ingressats a l'UCI i el model, en canvi, farà una predicció **un cop el pacient hi hagi ingressat.**

Per aquesta raó, s'han eliminat totes les files amb el contingut "UCI_DIAS = 0", que resulta en un total de **118** registres. Com que es tracta d'una quantitat molt petita de mostres, s'ha decidit canviar els percentatges del *Train* i *Test Set*, que han sigut 70% i 30%, respectivament.

Els valors de la columna son numèrics, ja que representen un nombre de dies; al tractar-se d'un model classificador, abans s'han d'agrupar els valors en diferents classes. Per fer-ho s'ha generat un histograma on visualitzar el repartiment dels dies (veure Il·lustració 61):



Il·lustració 11. Histograma de la columna "UCI_DIAS"

Es pot veure com la majoria de les dades estan concentrades a l'inici, entre els 0 i 10 dies. Com que no hi ha una separació clara entre elles, s'ha decidit provar agrupacions diferents, i igual que el model anterior, algorismes diferents per a cadascun d'ells (RF, SVM i KNN).

Algorisme 1: *Random Forest*

Per a predir les dades, la columna "UCI_DIAS", s'ha agrupat en 4,3 i 2 grups. I per a tots ells s'ha fet servir l'algorisme *Random Forests* amb els paràmetres per defecte.

El funcionament segueix sent el mateix: cridar la funció, entrenar les dades i fer una predicció.

Prova amb 4 grups

A partir de l'histograma s'ha intentat fer una agrupació per tal d'aconseguir 4 grups per a guardar-los com a variables de sortida. Aquests son: 1 a 10 dies, 11 a 20 dies, 21 a 30 dies i més de 30 dies (veure Il·lustració 62).

```
for days in Y:
    if(days < 11):
        y_class_four.append("1-10")
    elif(days < 21):
        y_class_four.append("11-20")
    elif(days < 31):
        y_class_four.append("21-30")
    else:
        y_class_four.append("+30")
```

Il·lustració 12. Agrupació en 4 grups

La classificació entre aquests grups, ha resultat en la següent matriu de confusió:

Classe real	1-10	11-20	21-30	+30
1-10	25	2	0	0
11-20	6	0	0	0
21-30	1	0	0	0
+30	2	0	0	0

Classe predita

Els càlculs de les mètriques son:

Accuracy	**69'44%**
Precisió	**18'38%**
"1-10"	73'53%
"11-20"	0%
"21-30"	0%
"+30"	0%
Threat Score/IoU	**17'36%**
"1-10"	69'44%
"11-20"	0%
"21-30"	0%
"+30"	0%

Es pot veure com l'única classe que s'ha encertat ha sigut la primera, "1-10", degut a que la majoritat de les dades es troben concentrades en aquell grup.

Prova amb 3 grups

Igual que abans, el primer pas, ha sigut agrupar les dades de sortia en 3 grups diferents, aquest cop: 1 a 7, 11 a 25 i +25 (veure Il·lustració 63).

```
for days in Y:
    if(days < 11):
        y_class_three.append("1-10")
    elif(days < 26):
        y_class_three.append("11-25")
    else:
        y_class_three.append("+25")
```

Il·lustració 13. Agrupació en 3 grups

I al fer la predicció, aquesta és la **matriu de confusió** obtinguda:

Classe real				
	1-10	25	2	0
	11-25	7	0	0
	+25	2	0	0
		1-10	11-25	+25

Classe predita

Amb les mètriques:

Accuracy	**58'33%**
Precisió	**24'51%**
"1-10"	73'53%
"11-25"	0%
"+25"	0%
Threat Score/IoU	**23'15%**
"1-10"	69'44%
"11-25"	0%
"+25"	0%

La precisió i el *Threat Score*, han sigut exactament els mateixos per a la classe "1-10", ja que és exactament la mateixa agrupació que abans. Però les mitjanes han augmentat una mica, al dividir entre menys classes.

Prova amb 2 grups

Aquest cop, les agrupacions han estat formades per 2 grups, dividits en: 1 i 15 dies, i +15 dies (veure Il·lustració 64).

```
for days in Y:
    if(days < 16):
        y_class_two.append("1-15") # 1
    else:
        y_class_two.append("+15") # 0
```

Il·lustració 14. Agrupació en grups de 2

A través de la predicció s'ha generat la següent **matriu de confusió**:

Classe real		1-15	+15
	1-15	27	3
	+15	5	1

Classe predita

I les mètriques son:

Accuracy	**76'78%**
Precisió	**54'69%**
"1-15"	84'38%
"+15"	25%
Threat Score/IoU	**44'13%**
"1-15"	77'14%
"+15"	11'11%

Un altre cop, el grup que conté els valors "1-15 dies", és a dir, els primers de l'histograma, son els que més encerts tenen.

Algorisme 2: *Support Vector Machines* (SVM)

Per aquest algorisme, també s'han fet agrupacions en 4, 3 i 2 grups, exactament els mateixos que l'anterior. I els passos a seguir per fer les prediccions també ho son: cridar el model, entrenar-lo i demanar una predicció.

Prova amb 4 grups

La **matriu de confusió** generada és:

Classe real		1-10	11-20	21-30	+30
	1-10	27	0	0	0
	11-20	5	0	1	0
	21-30	1	0	0	0
	+30	2	0	0	0

Classe predita

I els valors de les mètriques:

Accuracy	75%
Precisió	**19'29%**
"1-10"	77'14%
"11-20"	0%
"21-30"	0%
"+30"	0%
Threat Score/IoU	**19'29%**
"1-10"	77'14%
"11-20"	0%
"21-30"	0%
"+30"	0%

Els resultats son molt similars a l'algorisme anterior, amb la classe "1-10" tornant a ser la que més encerts té.

Prova amb 3 grups

La **matriu de confusió** generada és:

Classe real	1-10	11-25	+25
1-10	25	2	0
11-25	6	0	1
+25	1	1	0

Classe predita

I els valors de les mètriques:

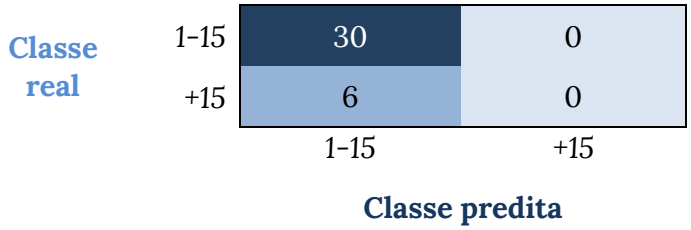
Accuracy	69'44%
Precisió	**26'04%**
"1-10"	78'13%
"11-25"	0%
"+25"	0%
Threat Score/IoU	**24'51%**

“1-10”	73’53%
“11-25”	0%
“+25”	0%

Igual que abans, els encerts es concentren en la primera classe, “1-10”.

Prova amb 2 grups

La **matriu de confusió** generada és:



I els valors de les mètriques:

Accuracy	**83’33%**
Precisió	**41’67%**
“1-15”	83’33%
“+15”	0%
Threat Score/IoU	**41’67%**
“1-15”	83’33%
“+15”	0%

Tornen a ser els mateixos resultats, en aquest cas, cap dels resultats ha donat “+15” com a possible classificació.

Algorisme 3: K-*nearest-neighbors* (KNN)

Finalment, s’ha fet servir l’algorisme *K-nearest-neighbors* per a contrastar els diferents resultats, sobretot, es vol veure com actua amb un conjunt de dades tant petit, ja que aquest algorisme està recomanat per aquests casos.

Com abans, es faran proves amb agrupacions de 4, 3 i 2 grups.

Prova amb 4 grups

La **matriu de confusió** generada és:

Classe real					
	1-10	27	0	0	0
	11-20	6	0	0	0
	21-30	1	0	0	0
	+30	2	0	0	0
		1-10	11-20	21-30	+30

Classe predita

I els valors de les mètriques:

Accuracy	**75%**
Precisió	**18'75%**
"1-10"	75%%
"11-20"	0%
"21-30"	0%
"+30"	0%
Threat Score/IoU	**18'75%**
"1-10"	75%
"11-20"	0%
"21-30"	0%
"+30"	0%

El mateix resultat que fins ara, a més, aquest cop, el model ha classificat a totes les entrades com a pertanyents a la classe "1-10".

Prova amb 3 grups

La **matriu de confusió** generada és:

Classe real				
	1-10	25	2	0
	11-25	6	0	1
	+25	0	2	0
		1-10	11-25	+25

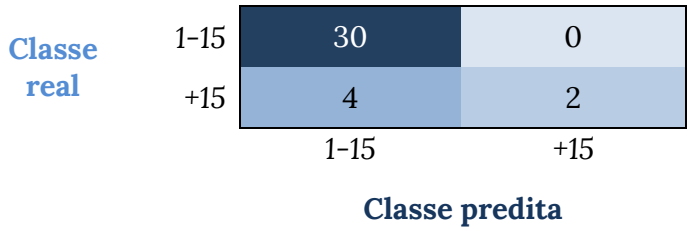
Classe predita

I els valors de les mètriques:

Accuracy	**69'44%**
Precisió	**26'88%**
"1-10"	80'65%
"11-25"	0%
"+25"	0%
Threat Score/IoU	**25'25%**
"1-10"	73'75%
"11-25"	0%
"+25"	0%

Prova amb 2 grups

La **matriu de confusió** generada és:



I els valors de les mètriques:

Accuracy	**88'88%**
Precisió	**91'12%**
"1-15"	82'24%
"+15"	100%
Threat Score/IoU	**60'78%**
"1-15"	88'24%
"+15"	33'33%

És el segon cop que s'ha obtingut un encert a fora de la primera classe, i en aquest cas, a més, la precisió ha sigut del 100%.

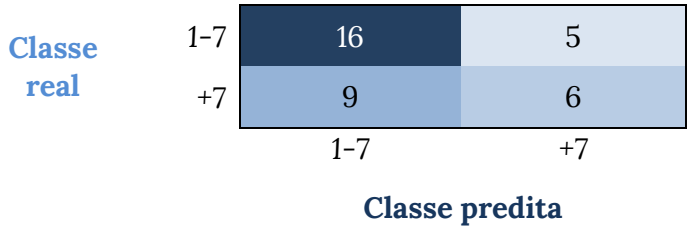
De tots els resultats obtinguts fins ara, el millor és amb: l'algorisme **KNN** amb agrupacions de **2 grups**.

Al haver observat que l'algorisme KNN ha donat els millors resultats, s'ha decidir fer proves agrupant diferents dies, per donar un rang més útil als metges, ja que en la classe "1-15", per exemple, hi ha massa distància entre el primer i últim valor. Per tant, s'han triat els grups descrits a continuació.

Nova agrupació 1 amb K-*nearest-neighbors* (KNN)

Aquesta nova agrupació consta d'una agrupació de **dos grups**: 1 i 7 dies, +7 dies; per donar una predicció als metges d'almenys una setmana o més.

La **matriu de confusió** generada és:



I els valors de les mètriques:

Accuracy	**61'11%**
Precisió	**54'40%**
"1-7"	59'26%
"+7"	54'54%
Threat Score/IoU	**41'67%**
"1-7"	53'33%
"+7"	30%

En comparació al grup de classes "1-15" i "+15", es veu que totes les mètriques han aconseguit un valor inferior, tant en les mitjanes com per classes.

Nova agrupació 2 amb K-*nearest-neighbors* (KNN)

Aquesta agrupació consta també té **dos grups,** ara son: 1 i 10 dies, +10 dies; per donar una predicció als metges d'almenys una setmana o més.

La **matriu de confusió** generada és:

Classe real			
	1-10	25	2
	+10	6	3
		1-10	+10

Classe predita

I els valors de les mètriques:

Accuracy	**77'78%**
Precisió	**70'33%**
"1-7"	80'65%
"+7"	60%
Threat Score/IoU	**51'51%**
"1-7"	75'75%
"+7"	27'27%

Aquest cop les mètriques han millorat una mica, però segueixen sent inferiors al grup inicial de "1-15" i "+15".

Nova agrupació 3 amb *K-nearest-neighbors* (KNN)

L'última agrupació consta **tres grups**: 1 i 7 dies, 8 i 14 dies i +14 dies, amb l'objectiu de millorar el resultat de l'anterior agrupació de 3 i donar als metges una predicció més específica.

La **matriu de confusió** generada és:

Classe real				
	1-10	17	0	4
	11-25	8	0	1
	+25	2	0	4
		1-10	11-25	+25

Classe predita

I els valors de les mètriques:

Accuracy	**58'33%**
Precisió	**35'8%**

“1-7”	62’96%
“8-14”	0%
“+14”	44’44%
Threat Score/IoU	**30’40%**
“1-7”	54’84%
“8-14”	0%
“+14”	36’36%

En comparació a l’anterior agrupació de 3, s’ha millorat la precisió i el *Threat Score*, però l’*accuracy* ha disminuït força, això vol dir, que encara que les classes estiguin més equilibrades, el model ha encertat menys valors en total. Per tant, s’hauria d’avaluar si es busca un equilibri o més encerts sense tenir en compte la classe.

5.1 Anàlisi de resultats

Els resultats de cada model s’han anat mostrant a mesura que s’ha fet cada prova. L’anàlisi de cadascun dels resultats, s’ha basat en la **matriu de confusió**, de la qual, s’han obtingut les tres mètriques per comprovar l’eficàcia del model. A continuació hi ha una guia dels resultats obtinguts:

Model 1

Prototipus	*Accuracy*	Precisió	IoU
RF amb dades inicials	76’29%	60’39%	38’16%
RF amb copies de dades	73’71%	56’46%	41’89%
RF amb SMOTE	71’83%	53’54%	40’61%
RF amb millors paràmetres + SMOTE	71’36%	52’95%	40’03%
RF amb millors paràmetres + SMOTE per a cada *fold*	68’31%	49’87%	37’48
SVM	67,84%	53’11%	40’32
KNN	63,15%	48’85%	35,64%
KNN + *Undersampling*	59’62%	50’31%	35’22%
KNN amb 2 classes (sense “NO ALTA”)	**82’80%**	**70’91%**	**58’89%**
RF amb 10 primeres variables més importants	75’12%	65’56%	35’95%

Si s'observen els percentatges, es veu clarament que el model prototipus amb millors resultat ha sigut el "**KNN amb 2 classes**", ja que només havia de classificar entre "Domicilio" i "Fallecimiento".

Ara bé, la tria del model depèn de si li donem valor a la classe "NO ALTA", ja que és important saber si un pacient seguirà ingressat o no, no només si morirà o serà enviat a casa. En aquest context, el que millors resultats ha obtingut, ha sigut el segon prototipus "**RF amb dades inicials**", sense la necessitat de cap modificació, tant en general, com a mitjana de classes.

Això ha demostrat que tot i que s'han provat diferents mètodes per a balancejar les classes, cap ha servit, ja que no aportava informació nova al model. Idealment, s'haurien de disposar de més dades de cada classe.

Model 2

Pel segon model, que servirà per una predicció sobre els dies que podria passar un pacient a l'UCI, també s'han fet diferents proves, aquí hi ha una guia dels resultats:

Versió	Grups	*Accuracy*	Precisió	IoU
RF: 4 GRUPS	1-10, 11-20, 21-30, +35	69'44%	18'38%	17'36%
RF: 3 GRUPS	1-10, 11-25, +25	58'33%	24'51%	23'15%
RF: 2 GRUPS	1-15, +15	76'78%	54'69%	44'13%
SVM: 4 GRUPS	1-10, 11-20, 21-30, +35	75%	19,29%	19,29%
SVM: 3 GRUPS	1-10, 11-25, +25	69,44%	24,51%	24,51%
SVM: 2 GRUPS	1-15, +15	83,33%	41,67%	41,67%
KNN: 4 GRUPS	1-10, 11-20, 21-30, +35	75%	18'75%	18'75%
KNN: 3 GRUPS	1-10, 11-25, +25	69,44%	25,25%	25,25%
KNN: 2 GRUPS	1-15, +15	**88,88%**	60,78%	**60,78%**
KNN: 2 GRUPS	1-7, +7	61'11%	54'40%	41'67%
KNN: 2 GRUPS	1-10, +10	77'78%	**70'33%**	51'51%
KNN: 3 GRUPS	1-7, 8-14, +14	58'33%	35'8%	30'40%

Aquí, la versió que destaca és la "**KNN amb 2 grups** [1-15, +15]", tot i que la mateixa versió amb els grups [1-10, +10] la supera en la precisió, majoritàriament la primera es millor.

A partir d'aquí, per triar un model, s'hauria d'estudiar quin rang de dies aporta més informació als metges.

Un radiòleg amb el qual es va posar en contacte un dels tutors, va recomanar l'agrupació [1-7, +7], però no ha proporcionat els millors resultats.

El problema ha semblat ser que la majoria de les dades estaven concentrades en els primers 10 dies, causant que el model no pogués diferenciar la resta de variables.

En resum, tot i que tots dos models tenen un prototipus amb bons resultats, si algú vol fer el seu ús, recomanaria fer un previ estudi de les seves necessitats.

5.1.1 Desplegament

Un cop acabat el procés d'entrenament i testeig, es té previst crear una **aplicació web** en local amb l'objectiu de mostrar una *demo* durant la presentació del projecte.

Aquesta aplicació consistirà en una pàgina web, on l'usuari podrà triar quin model vol utilitzar: per predir l'estat del pacient, o els possibles dies a l'UCI. A continuació, se li demanarà entrar les dades i es procedirà a fer la classificació i mostrar un resultat.

6 Conclusions i treball futur

6.1 Conclusions

El projecte ha consistit a crear dos models de *Machine Learning* amb l'objectiu principal d'ajudar a la comunitat sanitària i alleugerar la pressió mèdica amb la malaltia de la COVID-19.

El primer model, ajudarà a **predir l'estat futur** d'un pacient, mentre que el segon, predirà, entre diferents grups de dies, quants en podria **requerir un pacient a l'UCI.** Els resultats d'ambdós models han sigut satisfactoris, els millors prototipus han superat el 80% d'*accuracy*, és a dir, que han encertat més d'un 80% dels valors.

Un gran problema que ha tingut el procés d'entrenament ha sigut la diferència de presència de les classes en el conjunt de dades (la majoritària superava per 1000 files a les minoritàries). Això ha provocat que el model no pogués classificar reconèixer completament variables de classes petites.

S'han provat també diferents mètodes de balanceig de dades, on els resultats independents de cada classe no han millorat gaire, en alguns casos, fins i tot, la millor versió ha sigut el conjunt de dades original.

Tot i així, el projecte no està format només pels models, i es podria dir que s'han complert tots els objectius proposats:

- S'ha analitzat la base de dades.
- S'han après i entès mètodes d'IA i ML.
- S'han creat els dos models proposats i s'han validat en tot moment els seus resultats.
- I finalment, s'ha completat aquesta memòria.

Per tant, es podria dir que ha sigut un projecte força íntegre.

A més cada etapa del treball, ha aportat coneixements nous, no només el desenvolupament. Ha requerit fer una bona planificació del temps, aprendre a analitzar i entendre les dades amb les quals es treballaven i estudiar diferents algorismes per trobar-ne aquells que s'adaptin a les necessitats del projecte.

Ha sigut interessant també, observar com reaccionaven les dades amb diferents mètodes i diferents paràmetres, que han ajudar a tenir una visió del funcionament d'aquests algorismes, que s'han utilitzar mitjançant llibreries externes.

I sobretot, a part de conèixer la diversitat de mètodes que proporciona el camp de la Intel·ligència Artificial, ha ensenyat la seva importància i presència en les nostres vides, no només en el camp de la medicina, sinó a tot arreu.

Cal recordar que l'objectiu principal sempre ha sigut ajudar als sanitaris, que han treballat a primera línia durant aquesta pandèmia, i tant com si aquest projecte aporta valor al món real o no, espero que pugui servir com estudi, com a coneixement i com a mostra dels diferents recursos que es poden trobar a Internet per a crear qualsevol model que es vulgui des de zero.

6.2 Treball futur

Tot i considerar-lo un projecte bastant complert i haver-se assolit els objectius establers, hi ha aspectes que es podrien treballar i millorar, com ara:

Recopilació de noves dades

Com han mostrat els resultats, el model no ha tingut una precisió molt alta a l'hora de predir classes minoritàries. Per corregir això, seria interessant recopilar noves dades de pacient més variades, sobretot, de les classes que no tenen suficient informació.

Aquestes dades, es podrien aconseguir posant-se en contacte amb diferents hospitals per preguntar si estarien interessats en aportar les seves dades, sempre és clar, anonimitzades.

Utilització de nous algorismes

En aquest projecte s'han fet servir un total de tres algorismes per a entrenar els diferents models, i a gran escala, els resultats han sigut força similars. Per això, es podrien provar diferents mètodes d'aprenentatge supervisat existents i fer comparacions de les prediccions.

Estudi intensiu dels paràmetres

Les llibreries que ofereixen els mètodes d'intel·ligència artificial permeten també especificar els paràmetres d'entrada. En la majoria de prototipus del projecte, s'han fet servir els paràmetres per defecte. Una possible ampliació seria generar bucles amb diferents rangs per a trobar els valors que proporcionen una millor exactitud a les prediccions.

Desplegament per l'ús públic

Una altra ampliació del projecte, seria el seu desplegament *on-line* en una aplicació web pública, que permeti entrar variables clíniques per a que diferents usuaris hi puguin accedir per fer prediccions.

7 Bibliografia

Bhattiprolu, S. (15 d'abril del 2020). *Introductory Python tutorials for image processing.* YouTube. https://www.youtube.com/playlist?list=PLHae9ggVvqPgyRQQOtENr6hK0m1UquGaG

Brownlee, J. (15 de gener del 2020). *Random Oversampling and Undersampling for Imbalanced Classification.* Machine Learning Mastery. https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification/

Martin, D. (20 de maig del 2019). *How to do cross-validation when upsampling data.* KiwiDamien. https://kiwidamien.github.io/how-to-do-cross-validation-when-upsampling-data.html

Dwivedi, R. (13 de setembre de 2020). *How To Implement ML Models With Small Datasets.* AnalyticsIndiaMag. https://analyticsindiamag.com/how-to-implement-ml-models-with-small-datasets/

Singh Bisen, V. (9 de desembre de 2019). *Where Is Artificial Intelligence Used: Areas Where AI Can Be Used.* VSINGHBISEN. https://medium.com/vsinghbisen/where-is-artificial-intelligence-used-areas-where-ai-can-be-used-14ba8c092e73

(4 de setembre de 2020). *What is artificial intelligence and how is it used?.* EuropaRl. https://www.europarl.europa.eu/news/en/headlines/eu-affairs/20210603STO05416/plenary-highlights-covid-certificate-lux-prize-biodiversity

(Recuperat el març de 2021). *RandomForestClassifier.* Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

(Recuperat l'abril de 2021). *KNeighborsClassifier.* Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

(Recuperat l'abril de 2021). *Support Vector Machines.* Scikit-Learn. https://scikit-learn.org/stable/modules/svm.html

(Recuperat el maig de 2021),. *Under-sampling.* Imblearn. https://imbalanced-learn.org/stable/under_sampling.html

Arsene, C (8 d'abril de 2021). *Artificial Intelligence in Healthcare: the future is amazing.* Healthcare Weekly. https://healthcareweekly.com/artificial-intelligence-in-healthcare/

Tyagi, N (24 d'abril de 2020), *6 major branches od Artificial Intelligence (AI).* Analytics Steps. https://www.analyticssteps.com/blogs/6-major-branches-artificial-intelligence-ai

Adams, R.L. (Recuperat l'abril de 2021). 10 *Powerful Examples Of Artificial Intelligence In Use Today*. Forbes. https://www.forbes.com/sites/robertadams/2017/01/10/10-powerful-examples-of-artificial-intelligence-in-use-today/?sh=3a99f595420d

(20 de maig de 2020). *What is Ai? Ai Vs Machine Learning Vs Deep Learning.* Techno Beast. https://technical-beast.blogspot.com/2020/05/Artificial-intelligence.html

Peart, A. (29 d'octubre de 2020). *Homage to John McCarthy, the Father of Artificial Intelligence (AI).* Artificial Solutions. https://www.artificial-solutions.com/blog/homage-to-john-mccarthy-the-father-of-artificial-intelligence

Mardsen, P. (4 de setembre de 2017). *Artificial Intelligence Defined: Useful list of popular definitions from business and science.* DigitalWellbeing. https://digitalwellbeing.org/artificial-intelligence-defined-useful-list-of-popular-definitions-from-business-and-science/

Selvamanikkam, M. (20 d'agost de 2018). *Introduction to Artificial Intelligence.* Becoming Human. https://becominghuman.ai/introduction-to-artificial-intelligence-5fba0148ec99

(Recuperat l'abril de 2021). *The Three Types of Machine Learning Algorithms.* PioneerLabs. https://pioneerlabs.io/insights/the-three-types-of-machine-learning-algorithms/

Ray, S. (9 de setembre de 2017). *Commonly used Machine Learning Algorithms (with Python and R Codes).* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

Anurag, (17 d'agost de 1028). *Random Forest Analysis in ML and when to use it.* NewGenApps. https://www.newgenapps.com/blog/random-forest-analysis-in-ml-and-when-to-use-it/

8 Annex

1. Gestió del temps

1.1 Activitats

Les activitats del projecte corresponen gairebé completament als paquets de treball, on els apartats de sota (a l'esquema) serien com sub-tasques de cada activitat. A continuació es detallen cadascuna d'elles i la seva estimació de valors

1.1.1 Documentació

En el cas de la documentació, és una activitat que es fa durant tot el període que dura el projecte, ja que a la memòria s'especificarà cada procés de treball. Per tant, la seva estimació és igual a la duració del projecte.

1.1.2 Anàlisi de requisits

- Estimació optimista (tO) = 2 setmanes
- Estimació més probable (Tm) = 3 setmanes
- Estimació pessimista (tP) = 4 setmanes

Durada esperada (tE) = (2 + 4*3 + 4)/6 = 3 setmanes

1.1.3 Entrenament i pràctica

- Estimació optimista (tO) = 3 setmanes
- Estimació més probable (Tm) = 4 setmanes
- Estimació pessimista (tP) = 5 setmanes

Durada esperada (tE) = (3 + 4*4 + 5)/6 = 4 setmanes

1.1.4 Dades (normalització)

- Estimació optimista (tO) = 1 setmanes
- Estimació més probable (Tm) = 2 setmanes
- Estimació pessimista (tP) = 2.5 setmanes

Durada esperada (tE) = (1 + 4*2 + 2.5)/6 = 1,9 setmanes

1.1.5 Model

- Estimació optimista (tO) = 3 setmanes
- Estimació més probable (Tm) = 3 setmanes
- Estimació pessimista (tP) = 4 setmanes

Durada esperada (tE) = (3 + 4*3 + 4)/6 = 3,1 setmanes

1.1.6 *Testing*

- Estimació optimista (tO) = 2 setmanes
- Estimació més probable (Tm) = 3 setmanes
- Estimació pessimista (tP) = 3 setmanes

Durada esperada (tE) = (2 + 4*3 + 3)/6 = 2,8 setmanes

1.1.7 (a) Disseny del Front-End

- Estimació optimista (tO) = 1 setmanes
- Estimació més probable (Tm) = 1 setmanes
- Estimació pessimista (tP) = 1.5 setmanes

Durada esperada (tE) = (1 + 4*1+ 1.5)/6 = 1,1 setmanes

1.1.7 (b) Implementació del Front-End

- Estimació optimista (tO) = 1 setmanes
- Estimació més probable (Tm) = 1 setmanes
- Estimació pessimista (tP) = 1.5 setmanes

Durada esperada (tE) = (1 + 4*1 + 1.5)/6 = 1,1 setmanes

1.1.8 Desplegament

- Estimació optimista (tO) = 1 setmanes
- Estimació més probable (Tm) = 1 setmanes
- Estimació pessimista (tP) = 1.5 setmanes

Durada esperada (tE) = (1 + 4*1 + 1.5)/6 = 1,1 setmanes

2. Càlcul de pressupost

En aquest apartat es calcularà un pressupost aproximat seguint els paquets de treball i les hores previstes indicades als apartats anteriors. D'aquesta manera, es començarà calculant les hores totals treballades.

Tenim un total de **14 setmanes**, en les quals treballem 40h, per tant obtenim:

14*40 = 560h de treball

Ara que ja tenim les hores de treball, anem a desglossar els paquets i definir el preu/hora. Per fer-ho, he tingut en compte dos tipus de programadors: **programador de *Machine Learning*** (semblant a un analista de dades, solen cobrar més que programadors convencionals) i un **programador web** per la part del Front-End; aquest tipus s'especificarà a l'hora de fer el càlcul.

A més a més, he suposat que de 8h diàries treballades, unes 5h en son productives, és a dir, un **62'5%**, que també s'especificarà per a cada paquet de treball. En el cas de la memòria, com que es una tasca a fer durant tot el període, inclourem la seva dedicació en la resta d'activitats, i així, ens queden els següents càlculs:

A2: Anàlisi de requisits

- Costos directes [Part ML-Analista de dades] = 15€/h
- Costos indirectes [Baix] = 2€/h

Hores totals = 120h **Hores productives** = 75h
Total = 75h*17€ = **1275€**

A3: Entrenament i pràctica

- Costos directes [Part ML-Analista de dades] = 15€/h
- Costos indirectes [Mig] = 3€/h

Hores totals = 120h **Hores productives** = 75h
Total = 75h*18€ = **1350€**

A4: Dades

- Costos directes [Part ML-Analista de dades] = 15€/h
- Costos indirectes [Alt] = 5€/h

Hores totals = 80h **Hores productives** = 50h
Total = 50h*20€ = **1000€**

A5/A6: Model + *Testing* (simultàniament)

- Costos directes [Part ML-Analista de dades] = 15€/h
- Costos indirectes [Alt] = 5€/h

Hores totals = 120h **Hores productives** = 75h
Total = 75h*20€ = **1500€**

A7a: Disseny Front-End

- Costos directes [Part Dissenyador Web] = 13€/h
- Costos indirectes [Mig] = 3€/h

Hores totals = 40h **Hores productives** = 25h
Total = 25h*16€ = **400€**

A7b: Implementació Front-End

- Costos directes [Part Dissenyador Web] = 13€/h
- Costos indirectes [Mig] = 3€/h

Hores totals = 40h **Hores productives** = 25h
Total = 25h*16€ = **400€**

A8: Desplegament

- Costos directes [Part Dissenyador Web] = 13€/h
- Costos indirectes [Mig] = 3€/h

Hores totals = 40h **Hores productives** = 25h

Total = 25h*16€ = **400€**

Finalment, ja disposem del preu/hora per a cada paquet, i això ens permet fer una mitjana del cost de tot el projecte:

Preu/hora = 123 (suma de preus per hora)/7 (grups de setmanes) = 17'57€/hora
Hores productives totals = 350h

I amb aquests preus, el cost total del projecte queda a:

Cost total = 350h*17'57€ = 6149'50€