Sistema de predicció de tir en l'entorn de la NBA

Treball Final de Grau

Jordi Vilajosana Rodríguez

Director: Miquel Sànchez Marrè

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya

Especialitat: Computació

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

FIB

Resum

El món de l'esport professional cada cop recull més dades i informació de tot el que passa en les seves competicions. L'estudi d'aquestes dades pot donar un coneixement que ajudi a millorar el rendiment dels participants. En aquest treball ens centrarem en un esport concret: el bàsquet. Específicament en la lliga americana, la NBA. En aquesta lliga es recull moltes dades de totes les accions que succeeixen en un partit. En el nostre cas, ens centrarem en les accions de tir. D'aquestes accions es recull la informació de la posició del jugador, el tipus d'acció de tir, el temps restant, la distància del defensor, etc. A partir de totes aquestes dades volem crear un sistema predictiu de tir general mitjançant tècniques/algoritmes d'aprenentatge automàtic. Per aconseguir aquest objectiu hem treballat amb les dades per veure quines variables ens interessen o quines podem crear a partir de la informació existent. Per crear el millor model possible hem estudiat les diferents tècniques existents en l'entorn d'aprenentatge automàtic per problemes de classificació, i hem anat provant diversos enfocaments per obtenir una solució que considerem acceptable.

Abstract

Sports professional world collects more and more information about everything that happens in its competitions. The study of this data can give some knowledge that helps improving the performance of participants. In this project we will focus on a specific sport: basketball. More precisely in the american league, the NBA. These league collects data about many of the actions that happen during a game. In our case, we will focus on the shooting actions. From these actions we collect information about the position of the player, the type of shooting action, the remaining time, the defender distance, etc. From all these data we want to create a general shooting prediction system using machine learning techniques/algorithms. To achieve this objective we have worked with data to see which variables are more interesting or which variables we can create from existing information. In order to create the best possible model, we have studied the different existing techniques in the machine learning environment for classification problems, and we have tested different approaches to obtain a solution that we consider acceptable.

Índex

1-Introducció

1.1 - Context

Aquest projecte "Sistema de predicció de tir en l'entorn NBA" és un Treball de Fi de Grau (TFG) del grau d'Enginyeria Informàtica a la Facultat d'Informàtica de Barcelona [1] de la Universitat Politècnica de Catalunya [2]. S'utilitzaran els diferents coneixements aconseguits en les diferents assignatures d'aquest grau relacionades amb l'estadística, la intel·ligència artificial, l'aprenentatge automàtic i l'anàlisi de dades, entre altres.

L'objectiu d'aquest projecte és utilitzar eines d'aprenentatge automàtic per analitzar diferents dades esportives de bàsquet de la millor competició del món, la NBA [3], i poder desenvolupar un sistema de predicció de tir per a jugadors professionals. Aquest model pot ser utilitzat posteriorment per diferents professionals del sector per analitzar quines característiques poden millorar l'encert del tir per augmentar el rendiment del seu equip.

1.2 - Objectius

L'objectiu principal d'aquest projecte és el de desenvolupar un sistema de predicció de tir a partir de dades relacionades amb el moment del tir obtingudes en el transcurs d'un partit de la NBA. Aquest sistema haurà de predir amb precisió si un tir encerta o no a partir de la informació que li donem (posició, distància, temps restant, etc).

Per arribar a aquest objectiu es defineixen els següents sub-objectius:

- Identificar diferents zones d'un camp de bàsquet que faciliten/dificulten l'encert del tir.
- Identificar factors situacionals (temps restants, defensor, etc) que faciliten/dificulten l'encert del tir.
- Provar diferents algorismes per tal d'identificar quin és el millor pel nostre problema en concret.
- Validar correctament el model amb els mètodes adequats per minimitzar els problemes d'ajust (*"overfitting"* o *"underfitting"*).

1.3 - Actors

A continuació es defineixen els diferents actors implicats en aquest projecte, ja que són les persones o grups als quals va dirigit.

Professionals del bàsquet

Els diferents professionals del bàsquet seran els principals interessats en el projecte ja que seran qui el podran utilitzar per millorar el seu rendiment personal o del seu equip. Dintre d'aquests grup d'actors hi hauria jugadors, entrenadors professionals de bàsquet i tot el conjunt d'assistents del entrenador.

Periodistes

Un altre actor implicat serien els diferents professionals que analitzen el bàsquet per així tenir un millor coneixement dels patrons de tir dels diferents equips i poder-ne fer un estudi més detallat amb finalitats periodístiques.

Director del projecte

El director del projecte, Miquel Sànchez Marrè, és qui s'encarrega de guiar l'estudiant durant el procés de creació i desenvolupament per tal d'assolir amb èxit els objectius d'aquest projecte.

Desenvolupador

L'estudiant serà l'encarregat de desenvolupar el projecte i la persona amb més interès per assolir el millor resultat final possible.

2 - Abast i requisits del projecte

En aquest apartat explicarem riscos que poden ocorren que facin difícil aconseguir els objectius plantejats anteriorment. A la vegada, farem una explicació de la metodologia i el rigor que s'utilitzarà durant el desenvolupament d'aquest projecte i que creiem que ens ajudarà a assolir els objectius proposats. Però primer de tot definirem quins són els requisits que ha de tenir el nostre projecte.

2.1 - Requisits

Requisits funcionals

- El sistema ha de ser entrenable amb dades que tinguin les mateixes variables
- El sistema ha de predir correctament si un tir si un tir entra o no quan se li dóna informació de l'acció

Requisits no funcionals

- Tenir una precisió alta del model final
- El model ha de poder ser adaptable a situacions més concretes (com un equip o un conjunt de jugadors)
- Les dades han d'estar correctament preprocessades
- Analitzar les prediccions segons filtres concrets en les dades (zona de tir, moment del partit, distància, etc.)

2.2 - Riscos i obstacles

A continuació explicarem diferents riscos i obstacles que pot tenir el nostre sistema de predicció de tir.

Temps d'execució

En aquest treball utilitzarem un gran volum de dades per la creació del model (més de 200.000 observacions), per tant, el temps d'execució dels diferents algorismes durant l'entrenament del model pot arribar a ser excessiu, causant així un problema.

En cas de trobar-se amb aquest problema s'intentarà utilitzar tècniques de paral·lelització per millorar els temps d'execució o d'utilitzar un subconjunt de les dades per trobar els paràmetres òptims pels diferents algoritmes i tècniques.

Trobar el model òptim

Utilitzarem diferents algorismes d'aprenentatge automàtic, i cadascun d'ells amb diferents possibles paràmetres. Això pot ocasionar que no aconseguim trobar quin és el millor model possible o que el model final aconseguit no sigui suficientment bo.

Per solucionar aquest problema tindrem l'ajuda del director del projecte, el qual al tenir molta més experiència ens podrà assessorar per apropar-nos el màxim possible al òptim.

Variables ocultes

En la creació del model tindrem moltes observacions, i per cada observació tenim diferent informació (variables). És possible que hi hagi altres variables no disponibles que no es tinguin en compte les quals poden tenir una gran importància, fent que el nostre model no sigui suficientment bo.

S'intentarà obtenir el màxim d'informació possible per tenir un model el més complet possible.

Adaptabilitat del model

Un possible problema serà l'adaptabilitat del model a casos concrets. Utilitzarem un gran volum de dades intentant fer un model general, però això pot ocasionar que no es pugui adaptar fàcilment a casos concrets d'un jugador o equip.

Un altre problema de adaptabilitat serà d'aplicar-lo a altres lligues. Com hem dit, les dades són obtingudes de la NBA, la qual té unes característiques dels camps de bàsquet (distància línia 3 punts, altura cistella, etc) lleugerament diferents a les lligues europees o les lligues femenines.

Per això s'intentarà fer un model general el qual pugui ser posteriorment adaptat fàcilment a les necessitats particulars de cada equip o jugador, i el qual minimitza la importància de les diferències entre les diferents lligues professionals de bàsquet.

2.3 - Metodologia i rigor

2.3.1 - Metodologia SCRUM

Per la planificació d'aquest treball s'ha utilitzat un mètode àgil de cicles curts, concretament s'ha utilitzat la metodologia SCRUM [24]. Aquesta metodologia està organitzada en esprints o iteracions de curta durada (entre una i quatre setmanes), per tal de poder planificar projectes els quals poden estar subjectes a un gran nombre de canvis.

En l'inici de cada iteració hi ha una planificació del que es farà i al acabar s'avalua si s'han aconseguit els objectius proposats. Depenent del resultat obtingut en relació als objectius es planifica la següent iteració, prioritzant certes tasques com corregir errors que hagin sorgit. No totes les iteracions han de tenir la mateixa durada, es pot adaptar la llargada de cadascuna d'elles a les tasques que s'han de realitzar en aquell moment del projecte.

2.3.2 - Eines

Per la realització d'aquest projecte s'han utilitzat diverses eines de planificació i treball que ajudaran a facilitar el desenvolupament.

S'ha utilitzat el repositori GitLab [25] per tenir accés al codi des de qualsevol dispositiu, així com poder tenir un control de les diferents versions i canvis que es vagin produint en el projecte.

També s'han realitzat diferents reunions amb el director del projecte per presentar l'estat del projecte i realitzar consultes pel correcte desenvolupament del treball. Depenent de l'etapa del projecte i dels problemes que han anat passant s'han realitzat més o menys reunions i amb major o menor temps entre elles.

Per la gestió de les tasques i el seguiment de la metodologia Scrum es volia utilitzar l'aplicació Taiga [26]. Aquest software està centrat per utilitzar el mètode Scrum, i té diferents eines de seguiment de tasques, així com un control de les completades i les restants. Com explicarem més endavant, s'ha acabat optant per no utilitzar-la.

Aquestes eines ens han servit per fer un correcte seguiment de les tasques així com per validar si s'estan assolint els diferents objectius del treball.

2.3.3 - Validació

Al ser un projecte de recerca és molt important la validació de les dades durant el procés i de la correctesa dels resultats obtinguts en les diferents fases.

Una de les primeres parts del projecte consisteix en el pre-processament de les dades, durant el qual s'estudiaran les dades per evitar biaixos, *outliers*, valors perduts i altres problemes derivats de treballar amb un gran volum de dades.

Durant la realització dels experiments amb els diferents algorismes de aprenentatge automàtic i els diferents paràmetres de cadascun d'ells també és important seguir uns mètodes de validació per així trobar el millor model. Hem utilitzat la tècnica de *cross-validation* [27] per avaluar els resultats de l'anàlisi estadística i garantir la independència entre dades d'entrenament i test. Aquest mètode consisteix en realitzar diferents iteracions, en les quals es divideixen les dades en dos subconjunts (entrenament i test) diferent en cada iteració. En una iteració s'analitzen les dades del conjunt d'entrenament i es valida el resultat amb el conjunt de test. En *cross-validation* es realitzen diverses iteracions i es calcula la mitjana de les diferents mesures de predicció obtingudes (precisió i error) en cada iteració per obtenir el model més precís.

Per tant, utilitzant aquests mètodes de validació podem assegurar que el model final que hem trobat compleix amb les garanties necessàries de precisió i independència que volem en aquest projecte de recerca i així assolir els objectius proposats.

2.3.4 - Canvis en la metodologia

El principal canvi en el metodologia ha estat relacionat amb les eines utilitzades. Un cop començat el desenvolupament del projecte hem vist que utilitzar Taiga per controlar les tasques a realitzar no s'adapta a les nostres necessitats, per tant s'ha prescindit d'aquesta eina. Finalment s'han utilitzat eines més tradicionals com serien un bloc de notes i notes enganxables (*post-it)* per tenir un control de quines tasques s'havien de realitzar.

La pandèmia mundial i l'estat d'excepcionalitat en el que ens trobem també ha fet modificar la metodologia emprada finalment. Per una banda, la impossibilitat de realitzar reunions presencials amb el tutor ha fet que aquestes consistissin en l'intercanvi de correus electrònics o videotrucades. Per altre banda, al no poder desplaçar-se del domicili, la utilització de Gitlab per poder accedir al projecte des de diferents dispositius no ha estat necessari i per tant, s'ha utilitzat en menor mesura. També, el fet de tenir models d'una mida considerable (arribant als 2-3GBytes) portava problemes a la hora de tindre tot el projecte guardat a Gitlab. Per tant, s'ha utilitzat sobretot com a controlador de versions en local, però no com a repositori online.

3 - Estat del art

3.1 - Investigacions prèvies

El bàsquet professional, especialment la NBA, s'ha anat modernitzant i adaptant a les noves tecnologies existents. Entre aquestes millores tenim la recollida de dades constant durant el transcurs del partit, per tenir documentades la majoria d'accions que es duen a terme. En aquest projecte utilitzarem les dades disponibles referent a les accions de tir a cistella per a la creació d'un model predictiu de tir.

Hi ha hagut diversos articles de recerca relacionats amb el tema proposat en el nostre projecte actual. A continuació farem un petit recull de diversos d'ells, els seus objectius i les conclusions a les quals van arribar.

Ibañez [5] va analitzar l'eficàcia dels llançaments a cistella per trobar l'impacte de certes variables relacionades amb l'acció del llançament. Entre les variables a analitzar hi hauria el moment temporal dins el partit (quart i temps restant), pressió defensiva, zona del llançament, etc. Van realitzar una regressió logística multinomial a partir de 8.000 dades obtingudes en partits de temporada regular de la NBA. Van arribar a la conclusió que si hi havia diverses variables relacionades amb un major encert a cistella. Per exemple, els principis de partit on hi ha menys pressió defensiva o el estar més proper a cistella.

Ciampolini [6], seguint amb l'estudi mencionat anteriorment [5], van realitzar un estudi dels factors relacionats amb els tirs de camp encertats durant les finals de la NBA de 2014 ja que creien que al ser partits més importants al disputar-se el campionat trobarien diferències amb l'anterior estudi. Els seus objectius principals eren estudiar la relació de l'eficàcia del tir amb el nombre de passes durant la jugada del tir, el tipus d'ofensiva, i la condició del jugador (si estava defensat, distància del defensor, etc). Van arribar a la conclusió que la condició del jugador en el moment de tirar tenia una gran importància per predir l'encert del tir, especialment en jugades on el jugador no estava defensat o amb bastant espai per tirar.

Velazquez [7] va utilitzar tècniques d'aprenentatge automàtic per realitzar un estudi dels hàbits de tirs dels diferents jugadors de la NBA i la seva eficàcia segons la posició de la pista d'on tiraven. Va veure com diferents jugadors tenen hàbits de tir clarament diferenciables, amb diferents zones amb major eficàcia depenent del jugador.

Altres estudis relacionats amb la temàtica s'han centrat en estudiar les habilitats ofensives individuals dels jugadors [8], un estudi espaial dels tirs d'un jugador en concret [9] o un anàlisi del tipus de llançament segons el jugador o de la situació [10] .

Per tant, hem pogut veure com hi ha diferents treballs d'investigació similars al que estem proposant en aquest projecte i dels quals podem obtenir informació per millorar el nostre. Però el nostre projecte serà diferent, ja que volem crear *un model general de tir* el qual treballa amb *un volum de dades molt més gran* que els anteriors treballs (tindrem més de 200.000 dades). En el nostre treball no discriminem segons el moment de la temporada (regular o playoffs), tot i que estudiarem si tenen una influència en l'eficàcia. També estudiarem tots els tirs (encertats i fallats) i de tots els jugadors que van jugar durant el període en el qual s'han aconseguit les dades.

Per tot això, després de fer un estudi de la bibliografia existent relacionada amb la temàtica d'aquest projecte, es creu que aquest projecte és clarament diferents als anteriors esmentats i el qual pot proveir d'un coneixement diferent al obtingut en estudis previs. Tot i això, hi ha bibliografia suficient en la qual ens podem basar durant el desenvolupament d'aquest projecte.

3.2 - Algoritmes utilitzats

En aquest apartat farem un petit resum dels diferents algoritmes que hem utilitzat en la implementació dels nostres models de predicció. En aprenentatge automàtic depenent del tipus de variable a predir, es parla de sistemes de classificació o discriminació quan la variable a predir és categòrica, i de sistemes de predicció numèrica o de regressió quan la variable a predir és numèrica. En el nostre problema a estudiar estem davant d'un sistema de classificació, és a dir, intentem predir una variable categòrica amb dues categories (encertar, no-encertar/fallar). Els algoritmes utilitzats són:

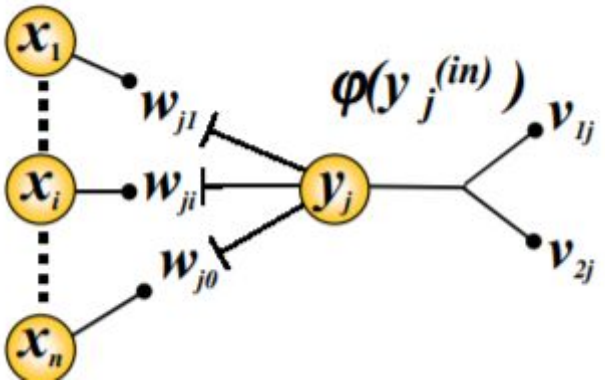
- Xarxes Neuronals Artificials
- Arbres de Decisió C4.5
- Support Vector Machines (SVM)
- K-Nearest Neighbours
- Ensemble Classifiers
    - Bagging
        - Random Forest
    - Boosting
        - Arbre de decisió C5.0
        - Gradient Boosting Machines
    - Stacking

3.2.1 - Xarxes Neuronals Artificials

Les Xarxes Neuronals Artificials (ANNs) són una tècnica d'aprenentatge automàtic utilitzada, entre altres, per problemes de classificació com el nostre. Les ANNs es basen en l'analogia existent entre el comportament i funcionament del cervell humà, el qual està format per neurones. Per tant, una ANN es caracteritza per tindre un conjunt d'unitats elementals amb baixa capacitat de processament, però que formen una densa estructura interconnectada amb un alt grau de paral·lelisme [11].

El funcionament d'una neurona (y_j) consisteix en què, a partir d'uns valors d'entrada (x_i) i uns pesos sinàptics (w_{ji}), se'ls hi aplica una funció d'activació per produir una o més senyals de sortida [12]. Una ANN està formada per una o més capes de neurones, en les quals la sortida d'una capa s'utilitza com l'entrada de la següent capa. A la Figura 1 es pot observar l'esquema d'una neurona artificial.

Figura 1: Neurona



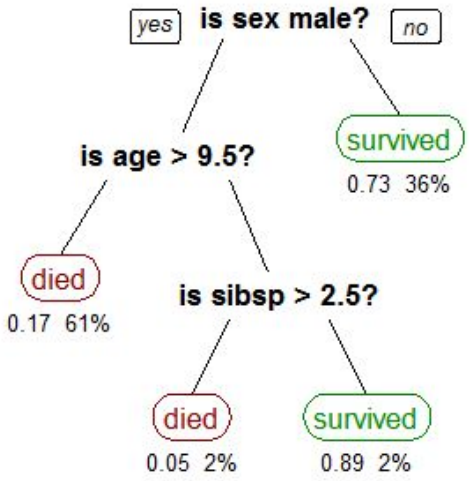
Font: [12]

La propietat més important de les ANNs és la capacitat d'aprendre a partir d'un conjunt de patrons d'entrenament, i així trobar un model que ajusti les dades [11].

3.2.2 - Arbres de decisió C4.5

L'aprenentatge basat en arbres de decisió són un conjunt de tècniques que utilitzen aquests com a model predictiu, especialment utilitzats en problemes de classificació. Una idea d'aquest algoritme la podem observar en la figura 2. En cada node de l'arbre s'utilitza un atribut de les dades per dividir aquestes en diferents subconjunts. Finalment, en les fulles de l'arbre tenim el resultat de la variable que volem predir. [13]

Figura 2: Arbre de decisió



Font: Wikipedia

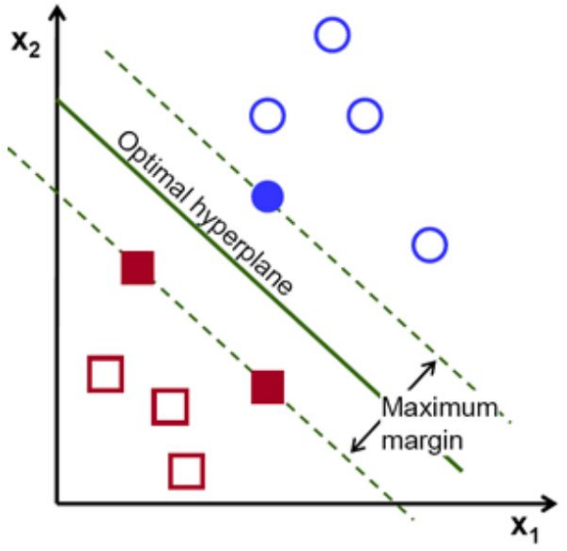
L'algoritme C4.5 (Quinlan, 1993) va ser creat com una millora del algoritme ID3 (Quinlan, 1986) per la creació d'arbres de decisió. El C4.5 crea un arbre de decisió realitzant particions de manera recursiva a partir de les dades d'entrada. A partir d'un conjunt de dades d'entrenament, en cada node de l'arbre el C4.5 tria l'atribut de les dades que divideix de manera més eficaç el conjunt de dades en valors diferents de la classe. El criteri per dividir les classes és la normalització del guany d'informació, és a dir, la diferència d'entropia que resulta de l'elecció d'un atribut per dividir les dades [14].

Com hem dit, el C4.5 va ser creat com a millora del ID3. Per exemple en els arbres C4.5 es poden utilitzar atributs continus i discrets, i també atributs amb costs diferents. Una altre millora és la post-poda, és a dir, un cop general l'arbre complet l'algoritme estudia quines branques es poden podar per millorar el rendiment i obtenir un arbre més compacte [15].

3.2.3 - Support Vector Machines

Les Support Vector Machines (SVM) són un conjunt d'algoritmes d'aprenentatge supervisat utilitzats per problemes de classificació i regressió. La idea bàsica d'aquest algoritme és trobar un hiperplà en un espai N-dimensional (on N és el nombre de variables) que classifiqui les diferents observacions. Perquè aquest hiperplà sigui l'òptim, les SVM utilitzen aquell que té el major marge possible, és a dir, que té la màxima distància possible entre observacions de les diferents classes [16]. A la Figura 3 podem veure un exemple d'una SVM.

Figura 3: Hiperplà òptim



Font: [17]

En problemes reals aquest hiperplà no serà un línia recta, ja que l'espai tindrà múltiples dimensions. Per solucionar aquesta problemàtica es poden utilitzar kernels. Un kernel és una funció que retorna el resultat del producte escalar entre dos vectors realitzat en un nou espai dimensional diferent a l'original [17]. Hi ha diferents tipus de kernels, per exemple lineals on K(x,y) = x*y o polinòmics llavors K(x,y)= $(x * y + c)^{d}$, entre molts altres.

3.2.4 - K-Nearest Neighbours

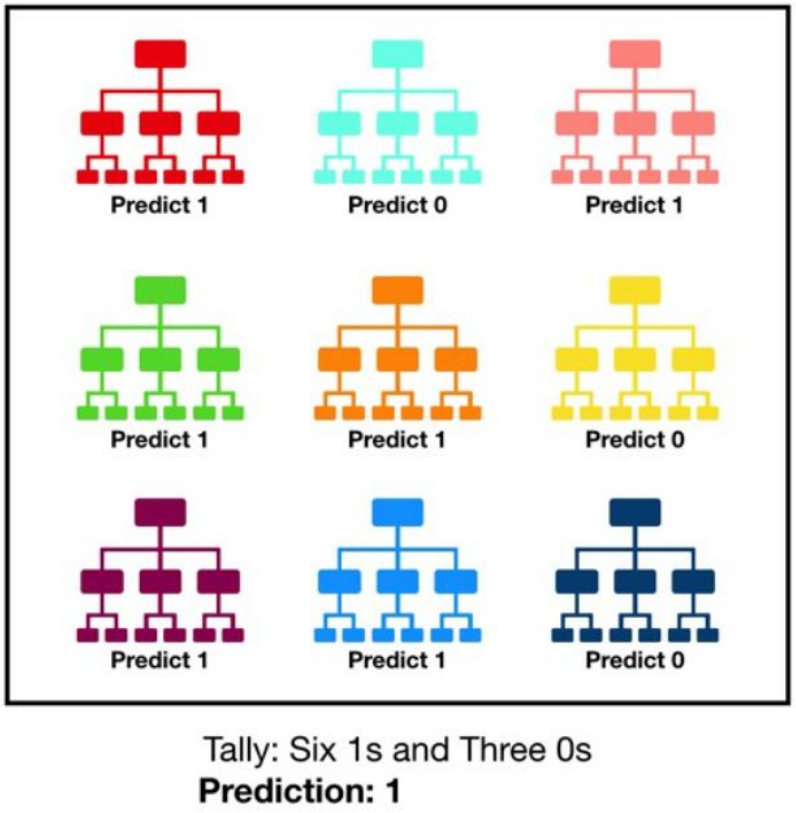
L'algoritme de K-Nearest Neighbours(KNN) és una particularització d'un classificador basat en casos, però és molt popular dins el conjunt d'algoritmes d'aprenentatge automàtic. La idea bàsica d'aquest algoritme és predir un valor buscant en les observacions més properes. Donat un element a predir, es calcula la seva distància respecte la resta d'observacions en les dades d'entrenament. Després es seleccionen els k elements més propers i segons la majoria d'aquests elements es decidirà la classificació final. Per mesurar la distància normalment s'utilitza la distància euclidiana o la distància del cosinus [18].

3.2.5 - Bagging i Random Forest

Els Random Forest (RFs) entren dins la categoria de tècniques de *bagging* sobre un conjunt de classificadors (*ensemble of classifiers)*. En un arbre de decisió normal, utilitzem unes dades d'entrenament de mida N, això provoca que un dels seus principals problemes és que són molt sensibles a les dades d'entrenament (*overfitting)*. En *bagging* s'agafen mostres aleatòries de mida N, però amb reemplaçament. L'altre característica d'aquests classificadors és l'elecció de la variable utilitzada per dividir un node. Anteriorment hem explicat que els arbres de decisió trien la que proveeix d'una millor separació. En els RFs aquesta elecció es fa d'entre un conjunt aleatori de variables. D'aquesta manera s'aconsegueix que hi hagi més variació entre els diferents arbres del model, i per tant, hi hagi una menor correlació entre ells [19].

Els Random Forests (RFs) són un tipus de conjunt de classificadors que construeix múltiples arbres de decisió durant l'entrenament, donant com a resultat la moda de les classes o la mitjana de les prediccions, segons si el problema és de classificació o de regressió [19]. En la Figura 4 podem veure representada la idea del RF per un problema de classificació. En aquest cas es construeixen 9 arbres de decisió diferents, i la predicció que realitza el RF és aquella que han fet més arbres.

Figura 4: Random Forest

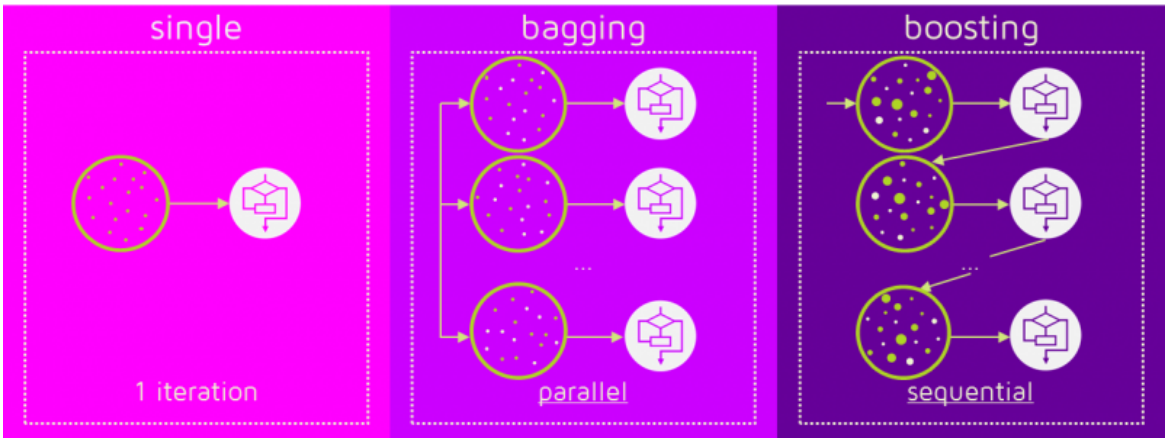


Font: [19]

3.2.6 - Boosting

Boosting és una tècnica basada en un conjunt de classificadors (*ensemble of classifiers*)format per un conjunt d'algoritmes que converteixen classificadors dèbils en classificadors robusts. La idea bàsica d'aquests algoritmes és entrenar seqüencialment aquests classificadors dèbils, de manera que cadascun intenta corregir al seu predecessor [20]. A la Figura 5 es pot observar les diferències entre les tècniques de *bagging* i *boosting*.

Figura 5: Diferència entre *bagging i boosting*



Font: [20]

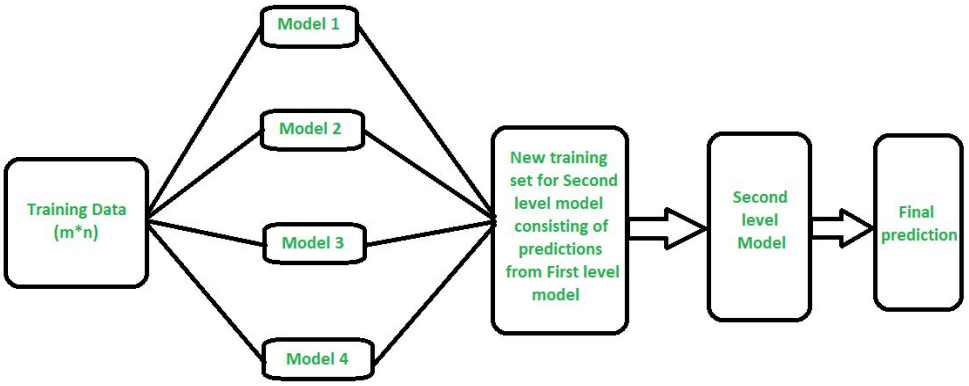
Dintre els diferents algoritmes de *boosting* nosaltres hem treballat amb les Gradient Boosting Machines (GBM). Aquest algoritme treballa seqüencialment afegint predictors al *ensemble*, de manera que intenta ajustar el nou predictor als errors residuals del predictor previ [20].

L'altre algoritme de *boosting* que hem utilitzat és el C5.0, el successor de l'algoritme d'arbres de decisió C4.5 explicat anteriorment. En aquest cas es generen diversos arbres de decisió, on cadascun d'ells es construeix prestant atenció als errors de predicció de l'anterior. Per tant, es van construint arbres de decisió diferents que intenten corregir els errors dels anteriors durant un nombre d'iteracions, o fins que s'ha aconseguit un classificador molt precís [21].

3.2.7 - Stacking

Stacking o Stacked Generalization és una tècnica de conjunt de classificadors (*ensemble of classifiers)* amb un enfocament diferent als vistos anteriorment. La idea d'aquesta tècnica consisteix en utilitzar diferents models per explorar un mateix problema, on cadascun d'ells aconseguirà aprendre una part del problema, però cap el problema complet. D'aquesta manera, entrenem diferents models per un mateix problema i després creem una predicció intermèdia, creada a partir de les prediccions de cadascun dels models individuals. Finalment, afegim un nou model que entrenem a partir d'aquesta predicció intermèdia [22]. A la Figura 6 es pot observar l'esquema de *stacking*.

Figura 6: Stacking



Font: [22]

Un cop fet un petit resum de la base teòrica dels diferents algoritmes que hem utilitzat, passarem a explicar les diferents eines que s'han utilitzat per implementar i validar aquests models.

3.3 - Tecnologies i eines utilitzades

En aquest apartat explicarem les diferents eines que s'han utilitzat per la implementació i validació dels diferents models i també alguna de les llibreries més importants que s'han utilitzat.

La principal eina pel desenvolupament d'aquest projecte és el llenguatge de programació i l'entorn integrat de desenvolupament (IDE) utilitzat. En el nostre cas ens hem decantat per utilitzar R i RStudio per sobre de Python. Al haver utilitzat R anteriorment en diferents assignatures, i especialment en l'assignatura d'Aprenentatge Automàtic, es tenia un bon coneixement del seu funcionament i de diferents llibreries relacionades amb aprenentatge automàtic.

Però la eina més important per la realització d'aquest treball ha estat el paquet caret (Classification and Regression Trainning) de R. Aquest paquet conté un conjunt de funcions que hem utilitzat per entrenar els diferents models, a la vegada que integra els diferents algoritmes que utilitzem. És especialment important ja que, amb un únic paquet integrem l'entrenament dels models, la validació, el test de diferents paràmetres, etc.

Com hem esmentat durant el transcurs d'aquest escrit, una de les parts més importants és la validació dels models. La validació d'un model consisteix en comprovar que els resultats del nostre models són precisos i generalitzables a un conjunt real de dades. En el nostre projecte hem decidit utilitzar la validació creuada amb diverses iteracions, més coneguda com *k-fold cross validation.* Aquesta tècnica consisteix en dividir el conjunt de dades en K subconjunts. Posteriorment s'utilitzarà un d'aquests subconjunts com a dades de test i la resta (K-1) com a dades d'entrenament. Aquest procés es repetirà durant K iteracions, per finalment realitzar la mitjana aritmètica dels resultats obtinguts del error o precisió dels models [23].

Finalment, tenim les eines que hem utilitzat per optimitzar els temps d'execució durant l'entrenament i validació dels models. Per una banda, hem utilitzat subconjunts de les dades per intentar tenir una estimació del temps que trigaria l'entrenament del model amb totes les dades. En la majoria d'algoritmes hem fet proves amb 10.000, 20.000, 50.000 i 100.000 observacions abans de realitzar-ho amb totes les dades (recordem que són 219.240 observacions). Ha estat molt útil ja que en molts casos l'augment del temps d'execució respecte l'increment en el nombre d'observacions no era lineal, passant d'uns segons a diverses hores. Gràcies a això, s'ha pogut planificar les execucions amb tot el conjunt de dades en moments on no era necessari l'ordinador ja que, degut al següent punt que explicarem, es quedava quasi bloquejat durant l'execució de l'entrenament del model.

Per altre banda, hem utilitzat paral·lelisme per intentar reduir el temps d'execució. Per això hem utilitzat les llibreries parallel i doParallel de R. Gràcies a paral·lelitzar l'entrenament dels models hem aconseguit una reducció important dels temps d'execució. El problema d'això ha estat que, al estar-se utilitzant múltiples *cores* del nostre ordinador, durant l'execució d'algun dels algoritmes era difícil poder-lo utilitzar ja que teníem la CPU i la memòria a més del 90% d'ús.

4 - Anàlisi i tractament de les dades

Les dades que hem utilitzat provenen de la web d'estadístiques oficial de la NBA [28], tot i que s'han obtingut des de una segona web [29] ja que era més fàcil la seva recol·lecció en format CSV per tractar-les. S'han utilitzat dades de la temporada 2014-2015 ja que és la última temporada sencera on es recollia informació de certs factors importants en el moment del tir. Per exemple es recollia la distància del defensor, el temps de possessió restant, temps que té el jugador la pilota abans de tirar, etc. Aquesta informació extra, que no es recull des de principis de 2016, és suficientment important pel nostre treball com per escollir dades de fa 5 anys per sobre les de l'últim any. En total tenim 219.240 observacions, és a dir, informació de més de 200.000 tirs de jugadors de la NBA i 22 variables per cada observació, tot i com veurem més endavant, aquest nombre variarà.

4.1 - Variables

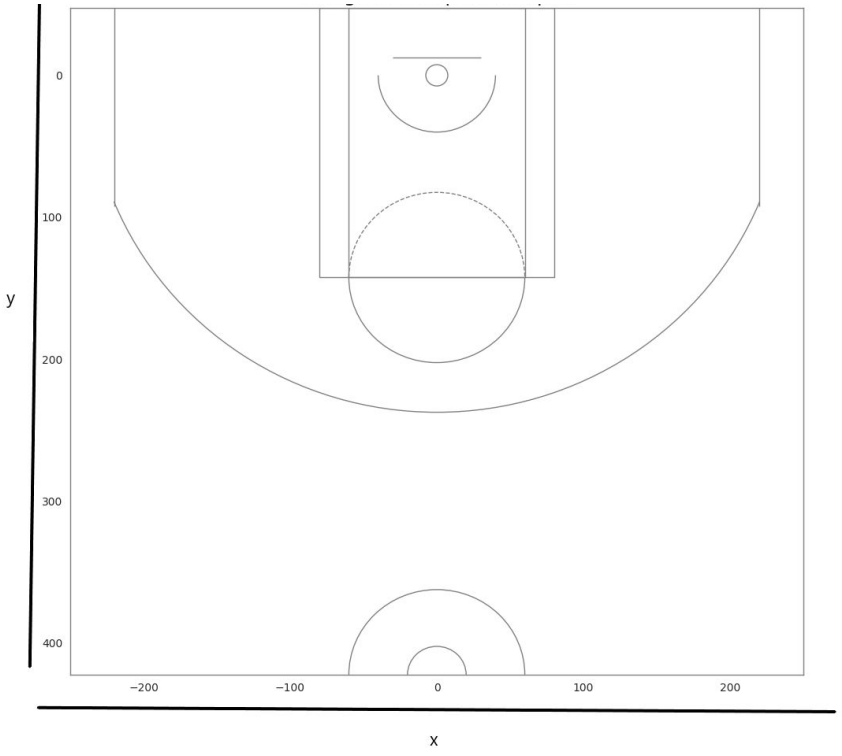
A continuació farem un llistat de les diferents variables que tenim inicialment, juntament amb una petita descripció de cadascuna d'elles.

- name: Nom del jugador que realitza el tir.
- team_name: Nom de l'equip al qual pertany el jugador.
- game_date: Data del partit on es realitza el tir.
- season: Temporada a la qual pertany la observació.
- espn_player_id: Número d'identificació únic per un jugador utilitzat per la cadena ESPN.
- team_id: Número d'identificació únic per un equip utilitzat per la cadena ESPN.
- espn_game_id: Número d'identificació únic per un partit utilitzat per la cadena ESPN.
- period: Quart/període del partit en el qual s'efectua el tir.
- minutes_remaining: Minuts restants en el quart/període.

- seconds_remaining: Segons restants en el quart/període.
- shot_made_flag: Recull si el tir ha estat encertat (1) o fallat (0).
- action_type: Tipus d'acció de tir.
- shot_type: Tipus de tir realitzat, segons si ha estat de 2 punts o un triple.
- shot_distance: Distància a cistella del tir realitzat.
- opponent: Equip rival en el partit.
- dribbles: Bots que ha realitzat el jugador en la jugada del tir.
- touch_time: Temps que ha tingut la pilota a les mans el jugador abans de tirar.
- defender_name: Nom del defensor en la jugada del tir.
- defender_distance: Distància a la que està el defensor.
- shot_clock: Temps de possessió restant en el moment de tirar.
- x: Posició X en el mig camp respecte la cistella.
- y: Posició Y en el mig camp respecte la cistella.

En la Figura 7 es veu un camp de bàsquet i com estan agafades les posicions x i y perquè sigui més fàcil d'entendre el seu valor.

Figura 7: Camp de bàsquet



Font: Elaboració pròpia

4.2 - Preprocessament de les dades

En aquest apartat parlarem de tot el preprocessament que hem fet a les dades. És un procés necessari ja que hem de netejar les observacions de valors nuls, erronis o *outliers*, eliminar variables que no aporten informació o afegir-ne de noves o transformar-les a partir de la informació que tenim.

4.2.1 - Variables afegides, transformades i eliminades

La primera transformació que vam fer, i la més simple, va ser passar les dues variables de distància al sistema mètric ja que estaven en peus (*feet)*. Tot i no ser un canvi estrictament necessari, ens serà més fàcil tractar i analitzar les dades si ho tenim en unitats internacionals, sobretot quan creem les zones de tir. Per fer aquest canvi simplement necessitem multiplicar el valor que tenim per 0.3048 [30].

El segon pas va ser crear una variable booleana anomenada "playoffs", la quan indica si el partit és de temporada regular (playoffs = FALSE) o a playoffs (playoffs=TRUE). Inicialment s'ha cregut que és interessant aquesta variable ja que en la post-temporada juguen menys equips, els quals son els millors, i a la vegada l'exigència i els nervis és major. Per tant, es creu que el fet de ser un tir en un partit de playoffs pot ser important, tal i com hem vist anteriorment en la investigació de Ciampolini [6]. Per calcular la nova variable s'ha consultat la data en que van començar els playoffs [31] i a partir d'aquí s'ha creat la nova variable a partir de la variable "game_date".

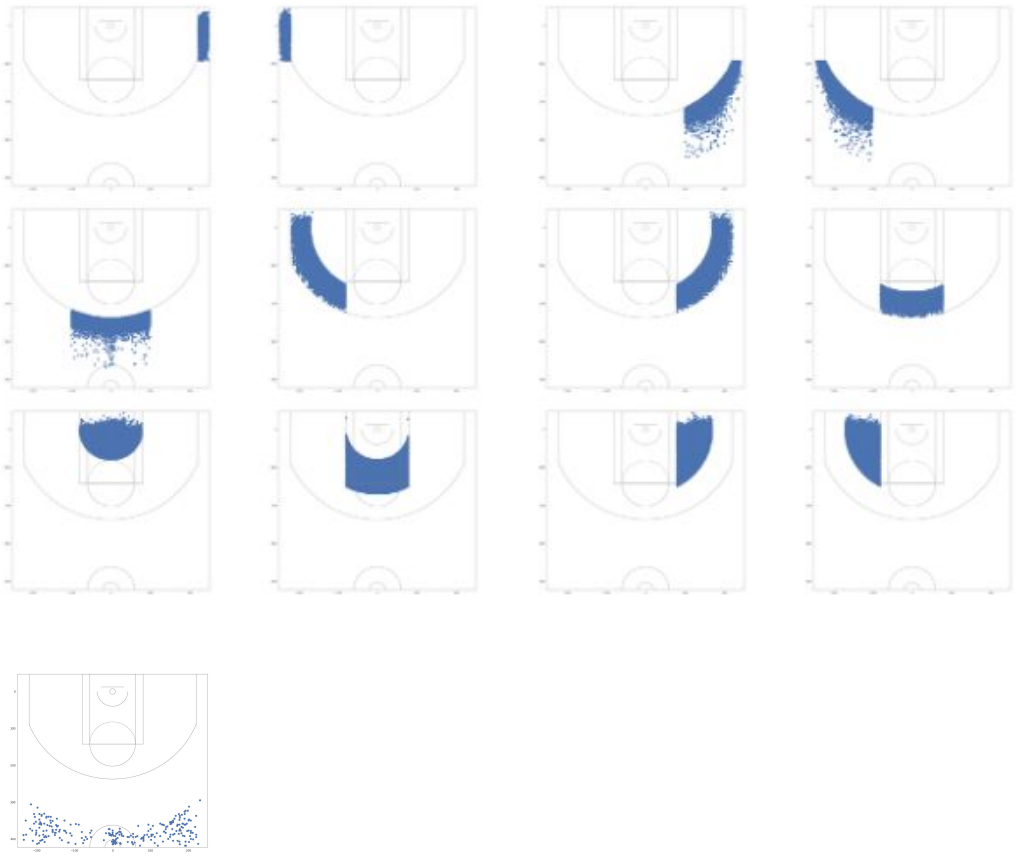
On hi ha hagut una major càrrega de treball ha estat per la creació de la variable "shot_zones", la qual a partir de la posició en el camp ("x" e "y") s'han creat diferents zones de tir. Hem creat un total de 13 categories per la nova variable, segons les diferents àrees de tir que enumerarem a continuació seguint l'ordre de la Figura 8:

1. 3PT right corner
2. 3PT left corner
3. 3PT right wing
4. 3PT left wing
5. 3PT frontal
6. 2PT left corner
7. 2PT right corner
8. Top of the key

9. Low post area
10. Free throw line area
11. Right short corner
12. Left short corner
13. Distant 3PT

S'ha decidit aquestes zones de tenir basant-nos en les que utilitza la web d'estadística de la NBA [28] i s'han calculat tenint en compte les variables "x" e "y", però també les variables "shot_distance" y "shot_type".

Figura 8: Zones de tir



Font: Elaboració pròpia

El següent pas era transformar la variable “action_type” ja que tenia un total de 49 categories, el qual és excessiu i un problema al treballar amb algoritmes com Random Forest. Per tant, s’ha creat la nova variable “action_type_shot”, a partir de la variable “action_type”, amb les següents categories:

- Jump Shot: Agafa totes les categories de la variable “action_type” on apareixen les paraules “jump shot”. Representa el que es considera un tipus de tir normal en bàsquet, és a dir, un tir amb salt.
- Layup Shot”: Agafa totes les categories on apareixen les paraules “layup shot”. Representa tots aquells dir on hi ha una entrada a cistella.
- Dunk Shot: Agafa totes les categories on apareixen les paraules “dunk shot”. Representa aquells tirs finalitzats amb una esmaixada a cistella.
- Hook Shot: Agafa totes les categories on apareixen les paraules “hook shot”. Aquesta categoria és per aquells tirs de “ganxo”, els quals són tirs a una mà amb un moviment d’arc que acaba per sobre el cap.
- Others: Agafa les categories que no s’han inclòs en alguna de les anteriors categories.

Finalment hem creat 3 variables fent referència als tirs realitzats i encertats per cadascun dels jugadors, i el seu percentatge d’encert. D’aquesta manera a partir de la variable “name” i “shot_made_flag” hem comptat el nombre total de tirs per jugador i el nombre de tirs encertats, i hem creat les variables “total_shots” i “shots_made”. A partir de la divisió “shots_made”/”total_shots” hem creat la variable “shot_percentage”. Això és important ja que per cada jugador tindrem la seva eficàcia en el tir, la qual hauria d’estar relacionada amb la probabilitat d’encertar el tir. I també és important tindre el nombre de tirs realitzats o encertats ja que no és el mateix un 50% d’encert tirant 4 tirs que tirant 1000.

Per acabar hem eliminat variables que no aporten informació o la que aporten ja la podem aconseguir d'alguna altre manera. Aquestes variables són les següents:

- "espn_game_id": És l'identificador del partit, no aporta ninguna informació.
- "team_id" i "espn_player_id": Són els identificadors per equip i per jugador, però aquesta informació ja la tenim en forma dels noms dels jugadors i els equips.
- "season": Fa referència a la temporada, però com tots els tirs són tots de la temporada 2014-15 és una variable constant per totes les observacions.

4.2.2 - Outliers, errors i valors nuls

Una part molt important del preprocessament de les dades és netejar-les per evitar tenir valors extrems (outliers), valors erronis o valors nuls. En aquest apartat farem un resum de les principals comprovacions que s'han realitzat per trobar i solucionar aquests problemes.

El primer pas va ser comprovar que no hi haguessin valors nuls (NA) en les nostres dades. Aquesta comprovació és molt fàcil de fer en R ja que només hem de fer un *summary* del nostre conjunt de dades i anar revisant les diferents variables. Vam veure que no hi havia cap valor nul en ninguna de les variables.

El segon que vam comprovar és que els tirs estiguessin ben categoritzats segons el seu tipus, és a dir, que els tirs de 2PT fossin dins la zona de 2 punts i els de 3PT fossin passada la línia del triple. Com podem veure en la Figura 1, vista anteriorment, la distància de la línia de triple no és la mateixa ja que no és un semicercle perfecte. Per tant la comprovació s'ha fet de dues maneres, per una banda s'ha mirat que els tirs de 3PT siguin a una distància superior als 6.7m i els de 2PT a una distància inferior als 7.24m, ja que són els dos límits de distància de la línia de triple segons si és frontal o lateral. Per altre banda, s'ha realitzat una comprovació gràfica, és a dir, dibuixar tots els tirs de 3PT i de 2PT per separat i veure que no hi hagués cap error.

En la comprovació més estadística hem comprovat que no hi ha cap tir de 2 punts ("2PT Field Goal") a una distància superior als 7.24m. Però al estudiar els tirs de 3 punts sí vam trobar un error, un tir en la categoria "3PT Field Goal" però la seva distància a cistella era inferior als 6.7m. Al ser només una observació es va estudiar el cas i es va veure que era clarament un error. Al estudiar l'observació es va veure que la distància i la posició ("x" e "y") corresponien a un tir de 2PT, i també l'acció de tir, ja que era un "Dunk Shot". Per tant, es va canviar el valor de l'observació a "2PT Field Goal" per corregir l'error.

La comprovació gràfica la podem observar a les dues següents figures. Es pot comprovar fàcilment com no hi ha més errors en la variable "shot_type" i que tots els tirs estan categoritzats correctament segons si són de 2PT o de 3PT.

Figura 9: Tirs de 3 punts

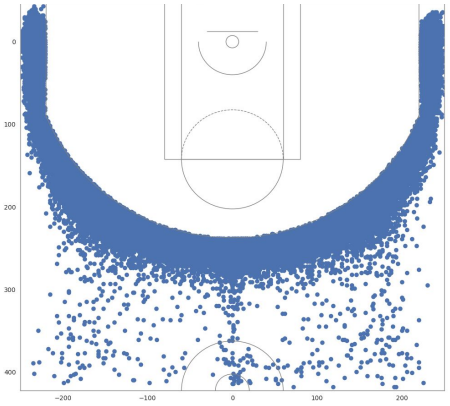
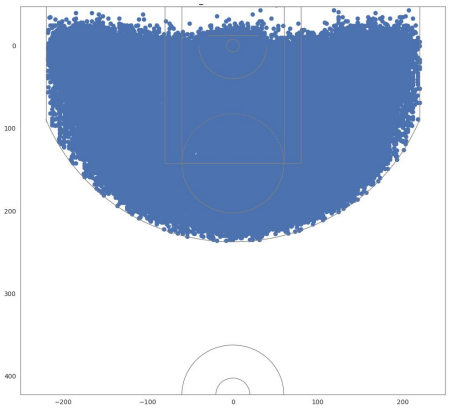


Figura 10: Tirs de 2 punts



Font: Elaboració pròpia

Una altre variable on vam trobar errors va ser en "touch_time". Al mesurar el temps que ha tingut la pilota a les mans un jugador, aquesta variable no pot tenir valors negatius. Vam observar com hi havia 472 casos negatius, els quals vam marcar com a valors nuls (NA), per tractar-los posteriorment.

Aquest han estat els errors que s'han trobat, a continuació farem un llistat de la resta de comprovacions que s'han realitzat en la cerca d'errors en les variables:

- "period": No pot ser inferior a 1, però pot ser superior a 4 ja que les prorrogues es compatibilitzen com període 5,6 i 7.
- "minutes_remaining" i "seconds_remaining": No poden quedar més de 12 minuts ni més de 60 segons. Tampoc poden tindre valors negatius.
- "shot_distance" i "defender_distance": No pot ser una distància negativa. Pels casos on sigui anormalment alta, ho veurem a continuació en el tractament d'outliers.
- "shot_clock": Ha de ser un valor entre 0 i 24, ja que és el temps màxim i mínim del rellotge de possessió.
- "dribles": No pot tenir un valor negatiu.

L'últim pas en la neteja de les dades ha estat la comprovació de l'existència *d'outliers*. Un outlier o valor atípic és una observació significativament diferent de la resta. En aquest projecte considerarem com outliers a tractar aquells que estiguin a una distància superior a 3 rangs interquartílics, és a dir, el que es coneixem normalment com outliers extrems. En aquests casos marcarem els valors com NA i posteriorment els tractarem.

Hem comprovat possibles outliers per les variables "shot_distance", "defender_distance", "touch_time" i "dribbles". La resta de variables numèriques, com "minutes_remaining" o "period", no les considerem en la detecció d'outliers ja que un valor atípic que no sigui un error pot ser perfectament normal i ens interessa mantindre. Per la única variable que no hem trobat valors atípics ha estat "shot_distance", per la resta sí hi ha havia valors atípics, els quals hem marcat com NA.

Finalment hem de tractar els casos NA. Tot i no haver-ne inicialment, degut a errors trobats i a valors atípics tenim diferents observacions amb algun valor nul en una variable. Per solucionar aquest problema hem imputat el seu valor amb l'algorisme de K-Nearest-Neighbours, el qual explicarem amb més profunditat posteriorment. Pel nostre cas, aquest algoritme omple els valors NA utilitzant una mitjana ponderada dels veïns més propers.

Un cop acabat el preprocessament de les dades procedirem a fer un estudi d'elles. Per una banda farem una selecció de característiques (*feature selection)* per escollir aquelles variables més importants i que utilitzarem en el modelatge. Per altre banda, realitzarem un anàlisi descriptiva de les variables més importants per entendre millor el nostre conjunt de dades.

4.3 - Estudi de les dades

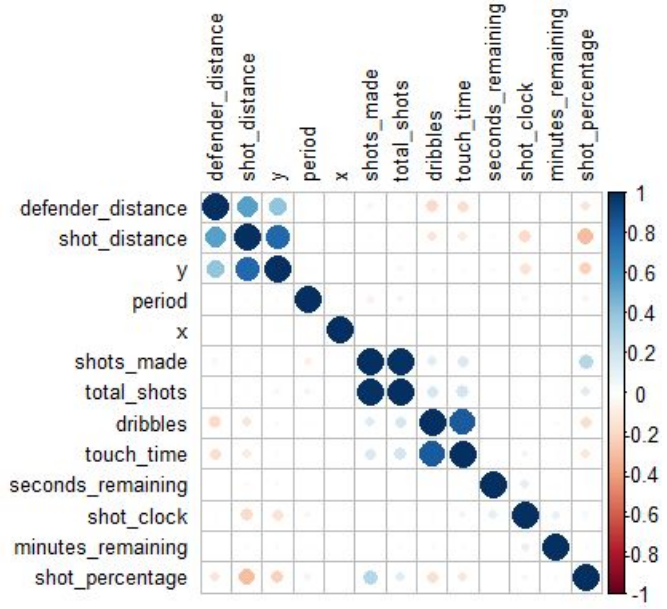
4.3.1 - *Feature Selection*

Una part important en el tractament de les dades és escollir aquelles variables que utilitzarem per entrenar els diferents models. Hem utilitzat 3 enfocaments per seleccionar les variables més importants. Primer de tot, nosaltres com experts hem descartat aquelles que no aporten informació rellevant al model. Segon, estudiant la correlació de les variables numèriques hem descartat aquelles amb una gran correlació entre elles. Finalment, hem utilitzat un mètode de feature selection anomenat Recursive Feature Elimination (RFE) per seleccionar les més importants.

Com hem dit, el primer pas ha estat eliminar les variables que nosaltres com experts sabem que no aporten informació rellevant al model. En aquest cas hem prescindit de les variables "name", "team_name", "opponent", "defender_name" i "game_date". Com es pot observar són variables per identificar la observació però que clarament no aporten informació important per l'objectiu del nostre model. A la vegada són variables amb moltes categories que donen problemes en algoritmes com el Random Forest.

Seguidament hem passat a intentar buscar variables redundants i eliminar-les. Per les variables quantitatives hem realitzat un anàlisi de la correlació entre variables. Per això hem creat una matriu de correlació, la qual podem observar de manera gràfica a continuació en la Figura 11:

Figura 11: Gràfic de correlació



Font: Elaboració pròpia

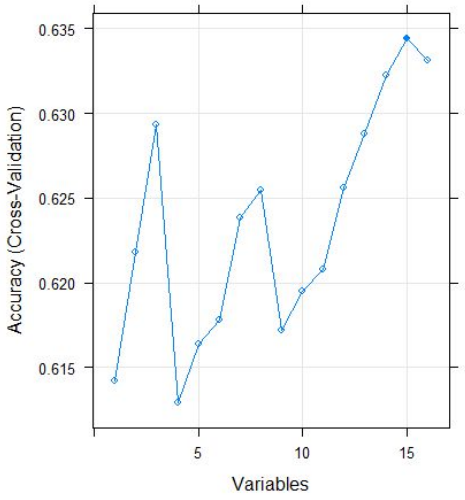
En aquest gràfic podem observar com hi ha 3 casos a estudiar detalladament, els quals són aquells amb una correlació superior a 0.75. Primer de tot tenim la relació entre "y" i "shot_distance", amb una correlació de 0.78. Tot i haver una gran correlació entre ambdues variables, seguirem mantenint ambdues variables per l'entrenament dels nostres models ja que ens aporten suficient informació. I tot i la gran correlació, al estar la variable "y" relacionada amb la variable "x" per identificar la posició en el camp creiem que és important mantenir-la en el nostre model. A la vegada, aquesta gran correlació pot ser donada per aquells tirs més llunyans els quals, com podem observar en la figura 8 anteriorment, augmenta molt el valor de "y" però no tant el de "x".

En segon lloc tenim la correlació entre “shots_made” i “total_shots”, la qual és de 0.98. En aquest cas és evident que mantenir les dos variables és redundant i per tant hem tret una del conjunt de variables que utilitzarem per entrenar els models. Hem decidit quedar-nos amb la variable “total_shots” ja que, de les dues variables, és la que té una menor correlació amb la variable “shot_percentage”, la qual s’ha creat a partir d’aquestes dues.

Finalment tenim el cas més problemàtic, i és la correlació de 0.84 entre “dribbles” i “touch_time”. Són dues variables que mesuren coses similars, ja que quants més bots que realitzi un jugador en teoria més temps tindrà la pilota a les mans per exemple. Però hem decidit quedar-nos amb elles perquè creiem que poden ser important els casos on no hi hagi relació entre les dues variables, és a dir, casos on hi hagi un valor alt de “touch_time” però baix de “dribbles” ja que podria significar, per exemple, una millor preparació del tir. També, com veurem a continuació, ambdues variables tenen importància per millorar la qualitat del model.

L’últim pas en la nostre selecció de variables pel modelatge ha estat aplicar el mètode de RFE. Aquesta tècnica consisteix en construir un model amb totes les variables, i l’algoritme va eliminant la pitjor variable d’una en una fins aconseguir el número desitjat [24]. En el nostre cas, com no sabem el nombre òptim de variables hem utilitzat cross-validation i hem provat amb totes les mides possibles.

Figura 12: Gràfic RFE



Font: Elaboració pròpia

Taula 1: Variables finals RFE

y	35.438285
x	30.033000
shot_distance	29.139018
action_type_shot	22.164448
touch_time	22.137794
shot_clock	19.200440
shots_made	17.094864
total_shots	16.071104
shot_zone	15.475136
dribbles	14.106153
shot_percentage	13.506957
shot_type	6.812645
defender_distance	6.151242
minutes_remaining	5.052694
seconds_remaining	4.024390

Font: Elaboració pròpia

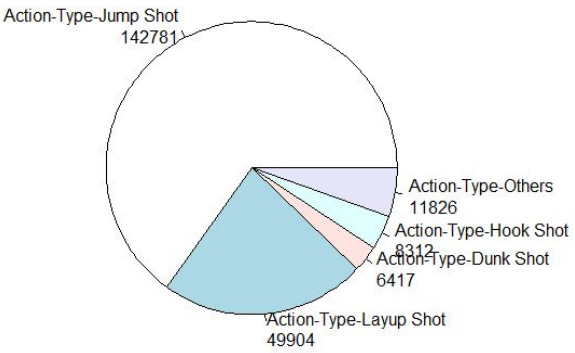
Com podem observar en la Figura 12 el nombre òptim de variables és 15, és a dir, només haurem d'eliminar una de les variables de les que ens queden. Aquesta variable, com podem observar ja que no apareix a la taula 1, és la variable "playoffs". És una variable que havíem creat nosaltres pensant que podria aportar informació útil pel nostre model, però ens vam equivocar. També podem observar com les variables de la posició "x" e "y" són les més importants pel model, o com les variables "touch_time" i "dribbles" tenen ambdues bastanta importància. Per tant, finalment ens quedem amb les 15 variables que apareixen a la taula 8 per entrenar els nostres models.

4.3.2 - Anàlisi descriptiva

Per finalitzar l'apartat de tractament de les dades realitzarem un anàlisi descriptiva d'algunes de les variables que tenim, especialment d'aquelles que considerem més importants. La més important és la variable que volem predir: "shot_made_flag". Al ser una variable binaria no podem extreure molta informació d'ella sola. L'únic que podem observar és com un 55.19% dels tirs són fallats ("shot_made_flag" = 0) i un 44.81% encertats. Per tant, si aconseguim un model amb una precisió similar o inferior al 55% serà un fracàs ja que seria la precisió que aconseguiríem si prediguéssim totes les observacions com a 0.

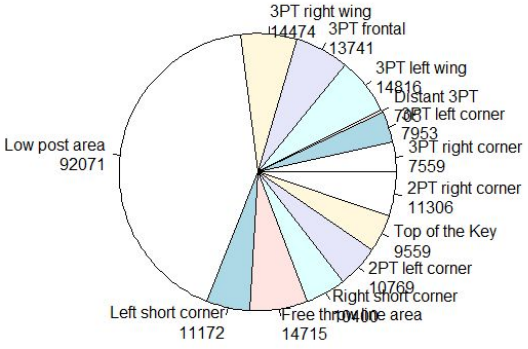
Dues variables que ens interessa veure una descripció del seu comportament són les dues creades per nosaltres: “action_type_shots” i “shot_zone”. A continuació podem veure gràfics de la distribució dels valors d’ambdues variables. Per la variable “action_type_shot” podem observar com un tir amb salt normal (“jump shot”) és la categoria amb la gran majoria dels casos amb un 65.12%. És un resultat normal ja que, tot i haver creat nosaltres les categories, en la variable original un “jump shot” bàsic ja era el 50% d’accions de tir. La segona categoria amb més observacions són les entrades a cistella (“layup shot”) amb un 22.76% dels casos.

Figura 13: Gràfic action_type_shot



Font: Elaboració pròpia

Figura 14: Gràfic shot_zone

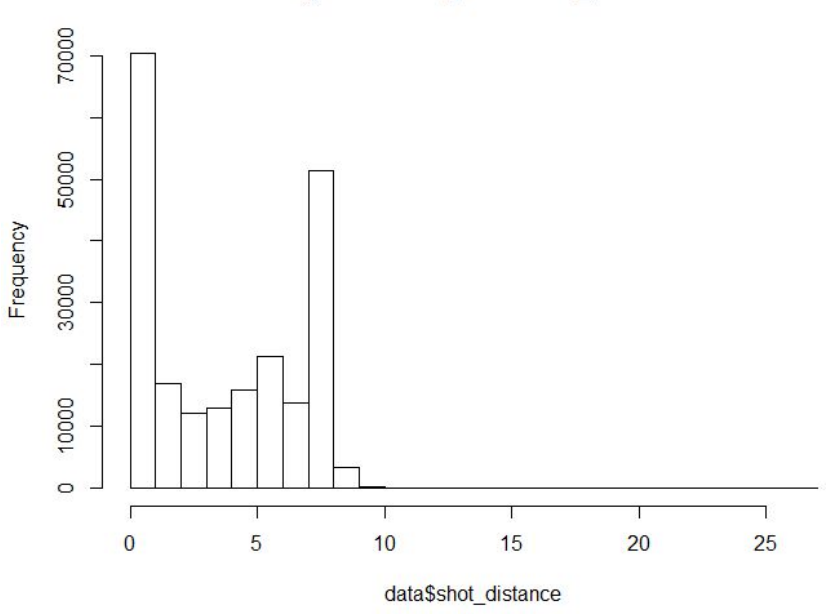


Font: Elaboració pròpia

Per la variable “shot_zone” observem un comportament similar. Per una banda, tenim que la zona més propera a cistella (“Low post area”) és la categoria amb més observacions amb un 42.00% dels casos. És normal ja que, en teoria, quant més a prop de cistella ets més fàcil ha de ser encertar el tir, a la vegada que la majoria de tirs després d’un rebot vindran d’aquella àrea. Per la resta de categories podem apreciar com hi ha una distribució bastant equitativa entre elles, tenint cadascuna d’elles entre un 3 i 7% de les observacions. La única categoria amb un nombre molt reduït de casos és la que fa referència als triples llunyans (“Distant 3PT”), la qual només conté un 0.32% dels casos. Tot i això, és normal que aquesta categoria tingui tants pocs casos ja que és un tipus de tir molt escàs i que en la majoria de vegades es recorre a ell per falta de temps de possessió.

Una de les variables que hem vist que té més importància és la distància des de la qual s'ha realitzat el tir ("shot_distance"). Com podem observar en el histograma que tenim en la Figura 15 es pot observar un comportament en forma de U en la distància. Tenim un pic principal pels tirs a una distància inferior a 1 metre, i un altre pic pels tirs a una distància d'entre 7 i 8 metres. Que els tirs de distància inferior a 1 metre siguin majoritaris té sentit ja que són els més fàcils de realitzar, a la vegada com gran part dels que venen posterior a un rebot. El pic d'entre 7 i 8 metres ve donat a que la distància de triple en la NBA són 7.24 metres, per tant en aquest rang és on es concentren la majoria de tirs de 3 punts. Observem com al augmentar la distància a cistella disminueix molt la quantitat de tirs, això és degut a què estàs augmentant la dificultat del tir al ser més llunyà, però no augmenta el benefici, cosa que passa quan arribes a 7.24 metres ja que cada tir val 3 punts i no 2.

Figura 15: Histograma shot_distance

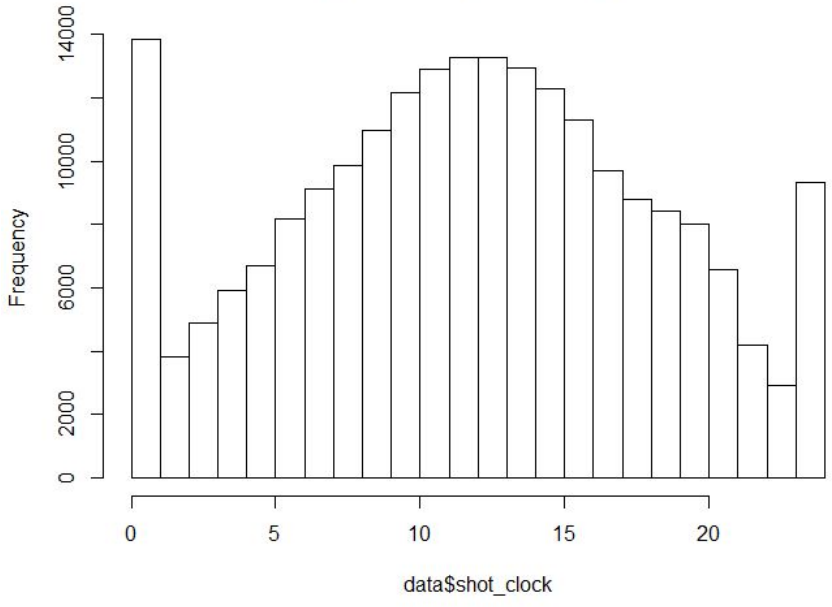


Font: Elaboració pròpia

Una altre variable important i que val la pena analitzar és el temps de possessió restant, és a dir, la variable "shot_clock". Com podem observar en la figura 16 a continuació, en aquest cas ens trobem que té una forma de W, amb un pic al principi i un al final, i al centre una certa distribució normal. Al igual que amb la variable "shot_distance", aquest comportament té una explicació senzilla. Que hi hagi una gran quantitat de tirs amb un temps de possessió inferior a 1 segon ve donat a que els jugadors en molts casos els jugadors s'esperen als últims moments de la possessió per realitzar el tir, així poden construir la millor jugada possible. O que hi ha una bona defensa que fa que no trobin una bona jugada i hagin de tirar al acabar-se la possessió.

En canvi, el pic al final, és a dir, tirs realitzats en el primer moment de la possessió és degut segurament a tirs després d'un rebot ofensiu. Quan un equip té la possessió i falla un tir i agafa el rebot, el rellotge de possessió es reinicia a 24 segons un altre cop[1]. Per tant, segurament aquests tirs són aquells que s'han realitzat després d'un rebot ofensiu o d'una situació que ha fet reiniciar el temps de possessió.

Figura 16: Histograma shot_clock



Font: Elaboració pròpia

[1] Des de la temporada 2018-2019 es reinicia amb 14 segons

Finalment, per acabar l'anàlisi descriptiva, veurem com es relacionen certes variables amb el nostre objectiu. El primer de tot seria veure les diferències en la distància del tir segons si el tir ha entrat o no. En aquest cas ens trobem com aquells tirs fallats tenen, de mitjana, una distància superior (4.31m) als tirs encertats (3.11m). Aquest fenomen és clarament comprensible ja que a mesura que t'allunyes de la cistella és més difícil encertar. També és comprensible que els tirs de dos punts tinguin un major percentatge d'encert (48.45%) que els triples (34.96%) ja que, en part, està relacionat amb la distància.

Una altre relació que ens interessa estudiar és la de la variable "shot_zones", ja que creiem que poden haver-hi zones amb major efectivitat. El primer que podem observar a la taula 2 és que la única zona amb un percentatge d'encert superior al 50% és la més propera a cistella ("Low post area"). També podem observar com la resta d'àrees tenen un percentatge similar (al voltant del 38-39%) excepte els triples llunyans, i els triples que no provenen de les cantonades. No tenim suficient informació com per explicar perquè els triples de les cantonades tenen un lleuger millor encert que la resta de triples, però podria ser perquè la distància és inferior, o perquè normalment són tirats per especialistes en el triple[2].

[2] És una suposició feta a partir de l'experiència com a jugador i espectador de bàsquet.

Taula 2: Relació zones de tir amb variable objectiu

Shot Zone	% Encert en la zona
3PT Right Corner	39.01%
3PT Left Corner	38.16%
Distant 3PT	2.69%
3PT Left Wing	34.18%
3PT Right Wing	34.32%
3PT Frontal	34.04%
Low Post Area	54.88%
Left Short Corner	38.96%
Right Short Corner	39.71%
Free Throw Line Area	41.61%
2PT Left Corner	39.00%
2PT Right Corner	39.12%
Top of the Key	39.47%

Font: Elaboració pròpia

Una altre variable que ens interessa estudiar com està relacionada amb l'encert del tir és el tipus d'acció ("action_type_shot"). En la taula 11 podem veure que els "Jump Shot" són l'acció amb un menor encert. És comprensible ja que, per una banda és el tipus de tir amb la majoria de casos, i per altre banda ja que la gran majoria de triples (els tirs amb menor percentatge d'encert) estan en aquesta categoria. També és important veure com les entrades a cistella ("Layup Shots") tenen un percentatge d'encert superior al 50%, però sobretot les esmaixades que tenen més del 90%. És clarament comprensible que tinguin tant alt encert ja que és una acció que consisteix, explicat de manera simplificada, en ficar la pilota dins la cistella mentre encara la tens a les mans.

Taula 3: Relació acció de tir amb variable objectiu

Action Type Shot	% Encert de l'acció
Jump Shot	36.24%
Layup Shot	57.73%
Dunk Shot	92.78%
Hook Shot	50.31%
Others	63.76%

Font: Elaboració pròpia

Per la resta de variables no s'ha observat diferències entre el percentatge de tir de les diferents categories, o la mitjana de la variable segons si era encert o fallada, que siguin significatius o importants pel nostre projecte.

5 - Construcció dels models i resultats obtinguts

En aquest apartat anirem parlant dels diferents models induïts a partir de les dades i dels resultats obtinguts. Com hem explicat en la planificació, després d'una reunió amb el tutor es va decidir provar amb els següents algoritmes: ANN, SVM, KNN, arbres de decisió C4.5 i Recursive Partitioning and Regression Trees (RPART). També es va decidir afegir els Random Forest, ja que tot i ser un conjunt de classificadors (*ensemble of classifiers)* és un dels algoritmes més coneguts i utilitzats en aprenentatge automàtic.

Els resultats de precisió i Kappa observats a continuació són obtinguts mitjançant *k-fold cross validation,* exactament amb 10 *folds,* tal i com hem explicat en la metodologia, i en tecnologies i eines utilitzades. També, quan considerem un model com a òptim al provar diferents paràmetres, és aquell amb una major precisió. Es seguirà un ordre cronològic en la presentació dels diferents resultats, és a dir, en l'ordre que es van anar induïnt els diferents models durant la realització d'aquest projecte.

5.1 - Xarxes Neuronals Artificials (ANN)

Les ANNs van ser el primer model que es va construir. Per aquest algoritme es va utilitzar la llibreria nnet dins el paquet caret. La parametrització d'aquest model es va fer mitjançant una matriu (grid) amb dos paràmetres: size i decay. El paràmetre size fa referència al nombre d'unitats dins la capa oculta, i el decay al paràmetre de regularització per evitar el *over-fitting.* S'ha fet un *grid* amb valors de decay de 10^{-5} fins a 10^{-1} (amb intervals de 0.2), i de size de 1 a 5, per un total de 105 combinacions. El valor òptim, és a dir, aquell amb major precisió, ha estat amb size = 4 i decay = 0.0002511886. En la Figura 17 podem observar un sub-conjunt dels resultats obtinguts al provar els diferents paràmetres. Al haver-hi 105 combinacions només s'ha inclòs el sub-conjunt on està inclòsa la combinació òptima.

Els resultats no van ser els esperats, ja que només es va aconseguir una precisió de 63.86% i un Kappa de 23.82%. El valor Kappa, o coeficient Kappa mesura la precisió però la normalitza segons la probabilitat aleatòria de les teves dades. Al tenir unes dades on per exemple es prediu tots els valors com fallats tindrem un 55% de precisió (ja que és el percentatge de tirs fallats totals), cosa que intenta normalitzar el coeficient Kappa. Per tant, també és important que ens fixem en el valor del coeficient Kappa per tindre una millor idea de la precisió real del nostre model.

Figura 17: Resultats de les ANNs

decay	size	Accuracy	Kappa
1.000000e-05	1	0.6296889	0.2283758
1.000000e-05	2	0.6346150	0.2304660
1.000000e-05	3	0.6360518	0.2329869
1.000000e-05	4	0.6360427	0.2336086
1.000000e-05	5	0.6378991	0.2378373
1.584893e-05	1	0.6238369	0.2057504
1.584893e-05	2	0.6331783	0.2285073
1.584893e-05	3	0.6362160	0.2336831
1.584893e-05	4	0.6370735	0.2365582
1.584893e-05	5	0.6376072	0.2366804
2.511886e-05	1	0.6232622	0.2057249
2.511886e-05	2	0.6264733	0.2091642
2.511886e-05	3	0.6367314	0.2357356
2.511886e-05	4	0.6371511	0.2355980
2.511886e-05	5	0.6367953	0.2353366
3.981072e-05	1	0.6276181	0.2234556
3.981072e-05	2	0.6346242	0.2298963
3.981072e-05	3	0.6354315	0.2326106
3.981072e-05	4	0.6367999	0.2354633
3.981072e-05	5	0.6379903	0.2370485
6.309573e-05	1	0.6204296	0.2156365
6.309573e-05	2	0.6345831	0.2317899
6.309573e-05	3	0.6361749	0.2341371
6.309573e-05	4	0.6372377	0.2358479
6.309573e-05	5	0.6377212	0.2369571
1.000000e-04	1	0.6229064	0.2180275
1.000000e-04	2	0.6351669	0.2324678
1.000000e-04	3	0.6363346	0.2340976
1.000000e-04	4	0.6361567	0.2343977
1.000000e-04	5	0.6379721	0.2372672
1.584893e-04	1	0.6289226	0.2267355
1.584893e-04	2	0.6349754	0.2328748
1.584893e-04	3	0.6367497	0.2350986
1.584893e-04	4	0.6368272	0.2346376
1.584893e-04	5	0.6373426	0.2357900
2.511886e-04	1	0.6262179	0.2224595
2.511886e-04	2	0.6333972	0.2301792
2.511886e-04	3	0.6356915	0.2332648
2.511886e-04	4	0.6385696	0.2382021
2.511886e-04	5	0.6376072	0.2367648
3.981072e-04	1	0.6195634	0.2010756
3.981072e-04	2	0.6330824	0.2301983
3.981072e-04	3	0.6356322	0.2318577
3.981072e-04	4	0.6372925	0.2358507
3.981072e-04	5	0.6298075	0.2149512

Font: Elaboració pròpia

5.2 - Support Vector Machines

Per els models de les SVM hem utilitzat 2 algoritmes, un amb un kernel lineal i l'altre amb un radial, amb la llibreria kernlab pels dos. En aquest cas ens hem trobat un problema important: el temps d'execució. Sense provar amb diferents paràmetres i amb totes les dades, el kernel lineal triga aproximadament 4 hores en acabar, i el radial el temps era superior a 8 hores. Per tant, les execucions si es provéssin diferents paràmetres es multiplicaria a més de 20 hores d'execució.

Figura 18: Resultats de les SVM

Model	Precisió	Kappa
SVM Linear	62.71%	22.95%
SVM Radial	62.98%	23.29%

Font: Elaboració pròpia

Finalment no li vam dedicar molt esforç i temps a aquests models donat que els resultats inicials tampoc eren esperançadors, ja que s'aconseguia una precisió al voltant del 63%. Podem observar els resultats obtinguts en la Figura 18. Donat aquests baixos resultats, i els alts temps d'execució, es va decidir passar a altres models que vam creure que podrien donar millors resultats.

5.3 - Arbres de decisió C4.5

El tercer model que es va provar van ser els arbres de decisió C4.5 mitjançant la implementació J48 dins el paquet RWeka. En aquest cas tenim 2 possibles paràmetres a ajustar: el factor de confiança C i el mínim nombre d'objectes en una fulla M. El factor de confiança influeix durant la poda de l'arbre, i representa el llindar d'error inherent que es permet en les dades mentre s'està podant l'arbre de decisió. Quant menor sigui aquest valor, es podarà més l'arbre i per tant, s'aconseguiran models més generals [33]. El nombre mínim d'objectes en una fulla serveix per obtenir models més simples i petits on les fulles tenen un gran nombre de mostres.

S'han fet diferents experiments amb els dos paràmetres C i M per intentar trobar el millor model possible. El paràmetre C té un rang de valors de 0 a 0.5, i M qualsevol nombre natural, però es recomana valors petits. Per defecte l'algoritme J48 en RWeka treballa amb C=0.25 i M=2. Després de diverses proves, totes elles amb un subconjunt de les dades, s'ha observat que l'increment de M fa decréixer la precisió, i el mateix amb el decrement de C. Per tant, els valors òptims pels diferents subconjunts han estat C=0.5 i M =1, tot i això, pel conjunt total de les dades s'ha provat amb diferents valors de C i M ja que el temps d'execució no era excessivament elevat. Finalment, s'ha fet la prova amb totes les dades amb una matriu (grid) on els possibles valors de M eren [1,2,] i de C [0.01, 0.5] per un total de 4 combinacions. En la Figura 19 observem els resultats obtinguts en l'execució amb tots els casos.

Figura 19: Resultats dels C4.5

C	M	Accuracy	Kappa
0.01	1	0.6435368	0.2480203
0.01	2	0.6433954	0.2477176
0.50	1	0.7192483	0.4174277
0.50	2	0.7120142	0.4024188

Font: Elaboració pròpia

Els resultats obtinguts finalment van ser iguals que amb els subconjunts, és a dir, els paràmetres òptims eren C=0.5 i M=1. Es va obtenir una precisió del 71.92% i un Kappa de 41.74%. Tot i no ser una gran precisió i Kappa, sí podem observar una gran millora respecte els models anteriors. Hem aconseguit una millora de un 8% respecte les ANN, però sobretot és important la millora en el Kappa, que ha estat del 18%, quasi el doble. No s'adjunta el arbre resultant ja que passat a format PDF ocupa un total de més de 400 pàgines, cosa que fa difícil el seu estudi.

5.4 - K-Nearest Neighbours

Després de la millora observada amb els C4.5, es va passar al model de raonament basat en casos utilitzant l'algoritme knn, de la llibreria knn. En aquest cas l'únic paràmetre disponible era K, és a dir, el nombre de veïns propers que utilitzarem per "votar" la classe d'un nou valor. Al ser un algoritme més ràpid s'ha optat per força bruta per trobar el valor òptim, per això s'ha anat fent proves amb diferents conjunts de valors (similar a una cerca dicotòmica) fins trobar la millor k, la qual ha estat 208. En la Figura 20 podem veure un resum de diferents valors que s'han provat per trobar la millor k.

Al ser l'algoritme més simple de tots, i donat els resultats obtinguts anteriorment, no teníem moltes esperances en aquest model. Finalment hem obtingut una precisió i Kappa molt similar al de les ANN, amb una precisió de 62.99% i un valor Kappa de 22.71%.

Figura 20: Resultats del KNN

k	Accuracy	Kappa
5	0.5794062	0.1421479
10	0.5914249	0.1621410
50	0.6225233	0.2135795
100	0.6279420	0.2227679
208	0.6299717	0.2273073
400	0.6299352	0.2284363

Font: Elaboració pròpia

5.5 - Recursive Partitioning and Regression Trees

Abans de començar amb els Random Forest es va intentar un model diferent d'arbres de decisió mitjançant l'algorisme rpart de la llibreria rpart. En aquest algoritme tornem a tenir només un possible paràmetre a determinar, el *complexity parameter* (cp). Amb aquest paràmetre podem controlar la mida de l'arbre per aconseguir l'òptima. D'aquesta manera si a l'afegir una nova variable al node actual té un cost superior al cp, s'atura la construcció de l'arbre. Aquest cost és la suma de les classificacions errònies a cada fulla, sumat a la multiplicació del nombre de particions per una penalització (lambda). Podem veure un resum de la fórmula en la següent figura 21:

Figura 21: Fórmula cost *complexity parameter*

$$\sum_{Terminal\ Nodes} Misclass_i + \lambda * (Splits)$$

Font: [34]

En aquest cas hem provat amb diferents valors de cp per trobar que l'òptim era un cp=0.0001, el qual és també el valor per defecte del algoritme en R. D'aquesta manera hem aconseguit una precisió i Kappa de 63.85% i 23.80%, les quals són molt similars a les obtingudes anteriorment amb els algorismes ANN i KNN. En la Figura 22 podem observar els resultats obtinguts amb diferents valors del paràmetre cp.

Figura 22: Resultats del RPART

cp	Accuracy	Kappa
0.0001	0.6385126	0.2379671
0.0101	0.6301656	0.2135832
0.0201	0.6301656	0.2135832
0.0301	0.6301656	0.2135832
0.0401	0.6301656	0.2135832
0.0501	0.6301656	0.2135832
0.0601	0.6301656	0.2135832
0.0701	0.6301656	0.2135832
0.0801	0.6301656	0.2135832
0.0901	0.6301656	0.2135832
0.1001	0.6301656	0.2135832
0.1101	0.6301656	0.2135832
0.1201	0.6301656	0.2135832
0.1301	0.6301656	0.2135832
0.1401	0.6301656	0.2135832
0.1501	0.6301656	0.2135832
0.1601	0.6301656	0.2135832
0.1701	0.6301656	0.2135832
0.1801	0.5519123	0.0000000
0.1901	0.5519123	0.0000000

Font: Elaboració pròpia

5.6 - Random Forest

L'últim model de la primera ronda van ser els Random Forest, els quals a diferència dels altres era un conjunt de classificadors(*ensemble of classifier).* Per aquest model hem utilitzat la implementació Parallel Random Forest (parRF), que utilitza les llibreries e1071, foreach, import i randomForest. El paquet caret té diferents implementacions de l'algorisme RF de diferents llibreries. Ens hem decidit per aquesta ja que, obtenint resultats molt similars entre les diferents implementacions, aquesta tenia un rendiment millor ja que requereix un menor temps d'execució.

En aquest cas el paràmetre amb el que podem treballar és el mtry. Aquest paràmetre fa referència al nombre de variables seleccionades aleatòriament com a candidats a utilitzar al dividir cada node. Hem realitzat els experiments del paràmetre amb un nombre reduït d'observacions (50.000 i 100.000) fins trobar el valor òptim. En la Figura 23 tenim el resultat obtingut en l'execució amb el millor valor del paràmetre mtry i tots els casos. En el nostre cas, el valor amb una millor precisió ha estat un mtry=4, amb una precisió de 63.94% i un Kappa de 24.19%.

Figura 23: Resultats del RF

mtry	Accuracy	Kappa
4	0.6394362	0.241921

Font: Elaboració pròpia

Per tant, vam acabar el primer conjunt de models obtenint uns resultats bastant pobres. La majoria dels resultats al voltant del 63% de precisió, amb només els arbres C4.5 aconseguint un 72%. Va ser en aquest moment que, donat els pobres resultats obtinguts, es va realitzar una altre reunió amb el tutor i es va començar a provar amb diferents conjunts de classificadors (*ensemble of classifiers)*.

5.7 - Boosting

Es va començar a provar els conjunts de classificadors (*ensemble of classifiers)* de la categoria *boosting*, concretament pels arbres de decisió C5.0 i els GBM. Es va decidir per aquests ja que, per una banda el C5.0 és una millora dels C4.5, els quals ens havien donat el millor resultat dels anteriors. Per altre banda, són el conjunt de classificadors (*ensemble of classifiers)* més simple, i per tant teòricament ens haurien de donar el pitjor resultat dels conjunts de classificadors (*ensemble of classifiers).*

Per l'algoritme C5.0 tenim els següents paràmetres: winnow, model i trials. El primer, winnow, és un booleà que indica si s'ha d'utilitzar *feature selection.* Els trials són el nombre d'iteracions de *boosting*, i el model indica si volem que sigui implementat mitjançant regles (*rules)* o arbres (*trees)*. El propi algoritme s'encarrega de provar les diferents combinacions per trobar la òptima. En la Figura 24 tenim les diferents proves de paràmetres que s'han realitzat. En el nostre cas el millor model ha estat aquell amb trials=1, model=tree i winnow=FALSE, obtenint una precisió de 64.77% i un Kappa de 25.60%.

Figura 24: Resultats dels C5.0

model	winnow	trials	Accuracy	Kappa
rules	FALSE	1	0.6432768	0.2467212
rules	FALSE	10	0.6379994	0.2471824
rules	FALSE	20	0.6384464	0.2508583
rules	TRUE	1	0.6409779	0.2414255
rules	TRUE	10	0.6370645	0.2437227
rules	TRUE	20	0.6372287	0.2444917
tree	FALSE	1	0.6477468	0.2560464
tree	FALSE	10	0.6458539	0.2552030
tree	FALSE	20	0.6439245	0.2558655
tree	TRUE	1	0.6431126	0.2464561
tree	TRUE	10	0.6415618	0.2450605
tree	TRUE	20	0.6400977	0.2447478

Font: Elaboració pròpia

Pel model fet amb les Gradient Boosting Machines també s'encarrega l'algorisme de testejar els paràmetres, que en aquest cas són: interaction.depth i n.trees. Interaction.depth és el nombre de divisions que fa en un arbre. I n.trees és el numero d'arbres. En la Figura 25 podem observar els resultats obtinguts. En el nostre cas el millor model ha estat aquell amb interaction.depth=3 i n.trees=150, amb una precisió del 64.03% i un Kappa de 24.03%.

Figura 25: Resultat dels GBM

interaction.depth	n.trees	Accuracy	Kappa
1	50	0.6314222	0.2174754
1	100	0.6334383	0.2229429
1	150	0.6344965	0.2265578
2	50	0.6346105	0.2251402
2	100	0.6372834	0.2323691
2	150	0.6388159	0.2367196
3	50	0.6368318	0.2302215
3	100	0.6399608	0.2385364
3	150	0.6403804	0.2403356

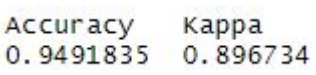
Font: Elaboració pròpia

Com podem observar en els resultats, no hem obtingut una millora important respecte els models anteriors. Especialment decebedor és el cas del model C5.0 el qual, al ser una millora dels arbres C4.5, pensàvem que obtindríem una major precisió, però no ha estat així. Els dos models realitzats amb el conjunt de classificadors (*ensemble of classifiers)* de *boosting* no han millorat el millor model trobat fins ara.

5.8 - Bagging

El següent conjunt de classificadors (*ensemble classifier)* que es va provar va ser un algorisme diferent de *bagging*, ja que, tot i els pobres resultats dels Random Forest, vam creure que un altre algoritme podria donar millors resultats. Després de fer una mica de recerca ens vam decidir per un Bagged Classification and Regression Trees (CART), exactament per la implementació treebag, la qual utilitza les llibreries ipred, plyr i e1071. Aquesta implementació no té paràmetres els quals podem variar per intentar millorar els resultats. En la Figura 26 podem observar els resultats obtinguts.

Figura 26: Resultats treebag



Accuracy	Kappa
0.9491835	0.896734

Font: Elaboració pròpia

Amb aquest model hem obtingut els millors resultats fins ara, amb una precisió de 94.91% i un Kappa de 89.67%. Això suposa un increment de més del 20% en la precisió respecte el millor model, però sobretot significa duplicar el valor Kappa del millor model. És especialment important haver aconseguit un valor Kappa molt similar a la precisió ja que, com hem explicat anteriorment, és el Kappa qui ens dóna una precisió més real del nostre model. Al igual que amb els arbres de decisió C4.5 el arbre resultant ocupa més de 200 pàgines en format PDF.

Figura 27: Variable importance

```
treebag variable importance

  only 20 most important variables shown (out of 45)

                                       Overall
x                                      100.000
seconds_remaining                       99.923
shot_clock                              97.495
shot_percentage                         91.065
touch_time                              90.703
defender_distance                       89.024
total_shots                             85.325
minutes_remaining                       70.453
shot_distance                           66.060
dribbles                                42.861
period                                  42.294
shot_zoneLow post area                  11.765
action_type_shotAction-Type-Dunk Shot   11.213
action_type_shotAction-Type-Layup Shot  11.030
shot_type3PT Field Goal                  7.913
action_type_shotAction-Type-Others       3.344
shot_zoneFree throw line area            3.209
shot_zoneTop of the Key                  2.547
action_type_shotAction-Type-Hook Shot    2.463
shot_zone2PT right corner                2.366
```

Font: Elaboració pròpia

En la Figura 27 podem observar un rànquing amb la importància de les diferents variables en la construcció del nostre model, concretament les 20 variables o categories amb major importància. Les variables amb major importància són la posició segons la x, el temps restant del període en segons, el percentatge de tir del jugador, el temps que ha tingut la pilota a les mans i la distància del defensor, totes elles amb una importància igual o superior a 90. Amb una menor importància, però dintre el top 20 podem observar diferents categories de dues variables que hem creat/transformat nosaltres com són “shot_zone” i “action_type_shot”. Per exemple la zona de “Low post area” o “Free throw line area”, o els tipus de tir d’esmaixades (Dunk shot) i entrades a cistella (Layup shot).

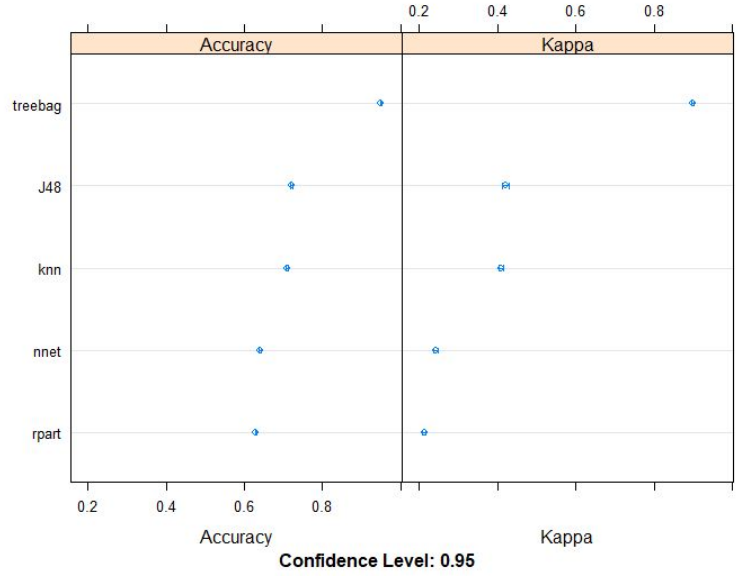
5.9 - Stacking

Tot i els bons resultats obtinguts amb *bagging*, volem millorar encara més el nostre model utilitzant el conjunt de classificadors (*ensemble of classifiers)* de *stacking.* Com hem explicat anteriorment, aquest model constarà de dues parts: primer, utilitzarem diferents models per crear una predicció intermèdia, i posteriorment entrenarem un model diferent basant-nos en aquesta predicció intermèdia. Per crear tots aquests models treballarem amb el paquet caretEnsemble, el qual conté diferents funcions que ens faciliten molt la feina. Per la primera part utilitzarem la funció caretList la qual, a partir d'un llistat de algoritmes de la llibreria caret, entrena els models i retorna una llista amb els models entrenats. Per la segona part utilitzarem la funció caretStack la qual, a partir d'un llistat de models entrenats, combina aquests models amb un nou algoritme per entrenar el model final, és a dir, realitza el *stacking* dels models individuals.

Pels algoritmes escollits per la primera part del *stacking* ens hem basat en dos criteris: resultats obtinguts i rendiment. Per una banda, volem aquells models amb els que hem obtingut uns bons resultats. Per altre banda, no volem aquells models que tenen un temps d'execució molt alt ja que farà que el temps total d'execució augmenti considerablement. Per això s'han escollit el algoritme de *bagging* anterior, els arbres C4.5, els RPART, les xarxes neuronals artificials i l'algoritme KNN. Per cadascun d'aquests models s'han agafat aquells paràmetres amb els que anteriorment hem obtingut millors resultats. S'han descartat els models de SVM ja que en les proves amb un nombre reduït de casos feien que augmentés considerablement el temps d'execució però no donaven una millora substancial al model final. Els Random Forest no s'han escollit ja que, al igual que les SVM, feien augmentar de manera important el temps d'execució final, i perquè s'utilitzaran com algoritme per modelar el segon nivell. Finalment, els arbres C5.0 no s'han inclòs ja que tot i ser una millora dels C4.5, no han donat millors resultats que aquests, per tant, hem inclòs els C4.5.

En la següent Figura 28 podem observar una comparació entre les precisions i valors Kappa dels 5 algoritmes escollits per realitzar *stacking*. Podem observar com hi ha 3 grups de resultats. Primer de tot tenim el treebag, l'algoritme de *bagging* amb un 94% de precisió. En segon lloc tenim els arbres C4.5 (J48) i els KNN, amb una precisió al voltant del 70%. Finalment, tenim els algoritmes de les ANN i RPART amb una precisió de 62-63%. És interessant veure gràficament l'evolució del Kappa ja que, tot i haver-ho comentat en els models, augmenta considerablement en relació amb la precisió.

Figura 28: Comparació models



Font: Elaboració pròpia

Finalment hem de realitzar el model de segon nivell, després de fer diferents proves ens hem decidit per un Random Forest. En aquest cas ens hem decidit per la implementació ranger, de les llibreries e1071, ranger i dplyr. Els RF són uns models que consumeixen molta memòria RAM durant la seva construcció, això provoca que altres implementacions (com la parRF utilitzada anteriorment) fallessin durant l'execució per falta de memòria. Per això s'ha escollit la implementació ranger, ja que hem trobat que s'ha pogut executar sense cap problema amb totes les dades i donant els mateixos resultats que altres implementacions al fer proves amb diversos subconjunts de dades.

Per aquest algoritme tenim 3 paràmetres: mtry, splitrule i min.node.size. El primer ja l'hem vist en el primer Random Forest i fa referència al nombre de variables seleccionades aleatòriament com a candidats a utilitzar al dividir cada node. La splitrule s'ha provat amb gini o extratrees. El gini fa referència a la impuresa de Gini i és una mesura de quan probable una classificació aleatòria d'un element del conjunt estaria incorrectament etiquetat si es classifiques de manera aleatòria segons la distribució del subconjunt. Per tant, amb aquesta splitrule es seleccionaria la divisió que minimitzés la impuresa de Gini. La splitrule extratrees, o *extremely randomized trees* és quan es selecciona aleatòriament tant la variable com el valor d'aquesta que s'utilitza en la divisió d'un node. Finalment, el paràmetre min.node.size és la mida mínima que poden tenir els nodes.

Figura 29: *Stacking* amb un RF

mtry	splitrule	min.node.size	Accuracy	Kappa
2	gini	1	0.9867360	0.9731566
2	gini	5	0.9778462	0.9551584
2	gini	10	0.9721082	0.9435428
2	gini	20	0.9669312	0.9330624
2	extratrees	1	0.9733443	0.9460379
2	extratrees	5	0.9661604	0.9315000
2	extratrees	10	0.9622560	0.9235988
2	extratrees	20	0.9593961	0.9178088
3	gini	1	0.9950602	0.9900115
3	gini	5	0.9894043	0.9785693
3	gini	10	0.9783981	0.9562890
3	gini	20	0.9705483	0.9403923
3	extratrees	1	0.9905902	0.9809647
3	extratrees	5	0.9744435	0.9482776
3	extratrees	10	0.9664888	0.9321705
3	extratrees	20	0.9615809	0.9222323
4	gini	1	0.9951058	0.9901037
4	gini	5	0.9914705	0.9827503
4	gini	10	0.9798577	0.9592461
4	gini	20	0.9716201	0.9425621
4	extratrees	1	0.9941662	0.9882031
4	extratrees	5	0.9774950	0.9544604
4	extratrees	10	0.9685778	0.9364018
4	extratrees	20	0.9626756	0.9244469
5	gini	1	0.9951286	0.9901499
5	gini	5	0.9924147	0.9846609
5	gini	10	0.9811348	0.9618322
5	gini	20	0.9720443	0.9434206
5	extratrees	1	0.9949279	0.9897444
5	extratrees	5	0.9789272	0.9573632
5	extratrees	10	0.9697318	0.9387388
5	extratrees	20	0.9634921	0.9261012

Font: Elaboració pròpia

El nostre model òptim, aquell amb la millor precisió i Kappa ha estat aquell amb un mtry=5, un min.node.size=1 i un splitrule seguint la impuresa de Gini, tal i com podem observar en la figura 29. Amb aquesta configuració hem aconseguit una precisió del 99.51% i un Kappa del 99.01%. D'aquesta manera aconseguim millorar la precisió aconseguida en el millor model anterior, el de *bagging*, en un 5% de precisió i un 10% de Kappa. Gràcies al conjunt de classificadors (*ensemble classifier)* de *stacking* hem aconseguit una precisió i Kappa superior al 99%, per tant, podem dir que hem aconseguit entrenar un model fiable que pot predir els tirs de bàsquet en la NBA.

5.10 - Resum

A continuació, en la Figura 30 tenim un resum dels millors resultats obtinguts en cadascun dels models. D'aquesta manera podem observar l'evolució que hem aconseguit amb els diferents models en la construcció del nostre sistema de predicció de tir.

Figura 30: Resum dels resultats

Model	Precisió	Kappa
ANN	63.68%	23.82%
SVM	62.98%	23.29%
C4.5	71.92%	41.74%
KNN	62.99%	22.71%
RPART	63.85%	23.80%
RF	63.94%	24.19%
C5.0	64.77%	25.60%
GBM	64.03%	24.03%
Bagging (treebag)	94.91%	89.67%
Stacking	99.51%	99.01%

Font: Elaboració pròpia

6 - Planificació Temporal

Segons la informació de la pàgina de la FIB relacionada amb el TFG [35], aquest té una càrrega de treball de 18 ECTS. Un crèdit ECTS és equivalent a 25-30 hores de càrrega de treball [36], per tant, aquest projecte ha de tenir una duració d'entre 450 i 540 hores.

Aquest projecte va ser pensat per començar el 17/02/2020 i finalitzar abans de la última setmana de juny. Per tant vam fixar com a data orientativa el 17/06/2020. Per tant, van ser aquest nombre de hores en aquest interval de temps el que es tindrà en compte a continuació per la planificació de les diferents tasques i el diagrama de Gantt.

A continuació farem un resum de la feina realitzada en les tasques realitzades i el treball previst en les que no s'ha començat.

6.1 - Tasques a realitzar

Aquest projecte l'hem dividit en 4 fases en les quals tindrem diferents tasques que explicaré detalladament. Aquestes 4 fases serien: Gestió del projecte, Investigació, Desenvolupament i Finalització.

Fase 1 - Gestió del projecte

L'objectiu d'aquesta fase és la planificació del projecte a realitzar. Aquesta fase s'ha realitzat en paral·lel amb la següent fase en alguns moments del projecte. A continuació s'explicaran les diferents tasques de les quals està formada aquesta fase.

Contextualització i abast (20 h)

En aquesta primera tasca es va definir el projecte i el seu abast. Vam decidir quins són els nostres objectius, actors implicats, riscos etc. I finalment es va definir l'abast del projecte.

Planificació temporal (15 h)

En aquesta tasca es va dur a terme la planificació temporal del projecte. Identifiquem les tasques a realitzar durant el projecte, la seva duració i com poden dependre unes de les altres.

Anàlisi econòmica i de sostenibilitat (20 h)

Aquesta tasca consisteix en la realització d'un pressupost per tal de dur a terme aquest projecte i en un anàlisi de l'impacte social, mediambiental i econòmic que podrà tindre el nostre projecte.

Fase 2 - Investigació

L'objectiu d'aquesta fase és la recerca d'informació i bibliografia sobre el tema del projecte. D'aquesta manera sabem quines investigacions prèvies hi ha hagut, quins mètodes han utilitzat i a quines conclusions han arribat.

Reunions amb el director (5 h)

Durant el transcurs del desenvolupament del projecte s'han realitzat diferents reunions amb el director del projecte. Inicialment van ser presencials, però donada la situació d'excepcionalitat actual s'han hagut de seguir fent via videotrucada o email.

Recerca d'informació (10 h)

Aquesta tasca va consistir en la cerca de diferents articles acadèmics i investigacions relacionades amb el objectiu d'aquest projecte o amb les tècniques que utilitzarem. Part d'aquesta tasca ja s'havia realitzat anteriorment en la contextualització del projecte.

Anàlisi de la informació (20 h)

Un cop obtinguda tota la informació es va fer un anàlisi més detallat d'aquesta per tenir una idea més detallada de quins algorismes utilitzarem, quines variables, quina validació, etc.

Aprenentatge (10 h)

Durant aquesta tasca ens vam familiaritzar amb el software i vam aprendre a utilitzar les eines que necessitarem posteriorment. Per realitzar aquest projecte utilitzarem R i Python, i tot i saber programar en aquests dos llenguatges, durant aquesta tasca vam investigar les diferents llibreries que hem acabat utilitzant durant tot el projecte.

Obtenció de les dades (5 h)

Per finalitzar aquesta fase, vam obtenir les dades les quals es hem fet servir durant el desenvolupament del nostre projecte.

Fase 3 - Desenvolupament

Aquesta és la fase principal del nostre projecte on hem estat desenvolupant el model i validant-lo per assolir els nostres objectius.

Pre-processament de les dades (50 h)

La primera tasca del desenvolupament va ser pre-processar les dades. És una tasca de vital importància ja que és on ens vàrem assegurar de no tenir valors perduts, outliers, errors, etc. I en els casos que vam tindre, es van prendre les mesures adequades per solucionar-ho. També vam realitzar diferents transformacions amb les variables per millorar la qualitat de la informació de la qual disposem.

Anàlisi descriptiva (20 h)

Aquesta tasca va consistir en l'anàlisi de les dades per tal de posteriorment utilitzar algorismes d'aprenentatge automàtic i crear un model.

Modelatge i entrenament (150 h)

Durant aquesta tasca vam utilitzar diferents algorismes d'aprenentatge automàtic per entrenar els diferents models. Hem utilitzat diferents models i provat diferents paràmetres fins a trobar el que té una major precisió.

Aquesta tasca ha consistit en diferents esprints on en cadascun d'ells s'ha creat un model amb un algorisme concret, per posteriorment entrenar-lo. D'aquesta manera hem anat provant diferents models fins a trobar el millor model possible.

Validació (100 h)

Aquesta tasca està altament relacionada amb l'anterior ja que consisteix en la validació dels models trobats. En aquest cas hem buscat millorar la precisió dels diferents models per aconseguir la millor possible.

Al igual que la tasca anterior aquesta ha consistit en diferents esprints on en cadascun d'ells s'ha realitzat la validació del model per trobar la màxima precisió.

Fase 4 - Finalització

En aquesta ultima fase es finalitzarà el treball, per tant, consistirà de les diferents tasques necessàries per acabar de perfeccionar l'entrega del projecte.

Redacció (60 h)

Aquesta tasca ha consistit en la redacció del document final a entregar. Ha estat en aquesta tasca on s'ha fet un recull de tots els models entrenats i validats, i de les conclusions a les quals s'ha arribat.

Defensa oral (10 h)

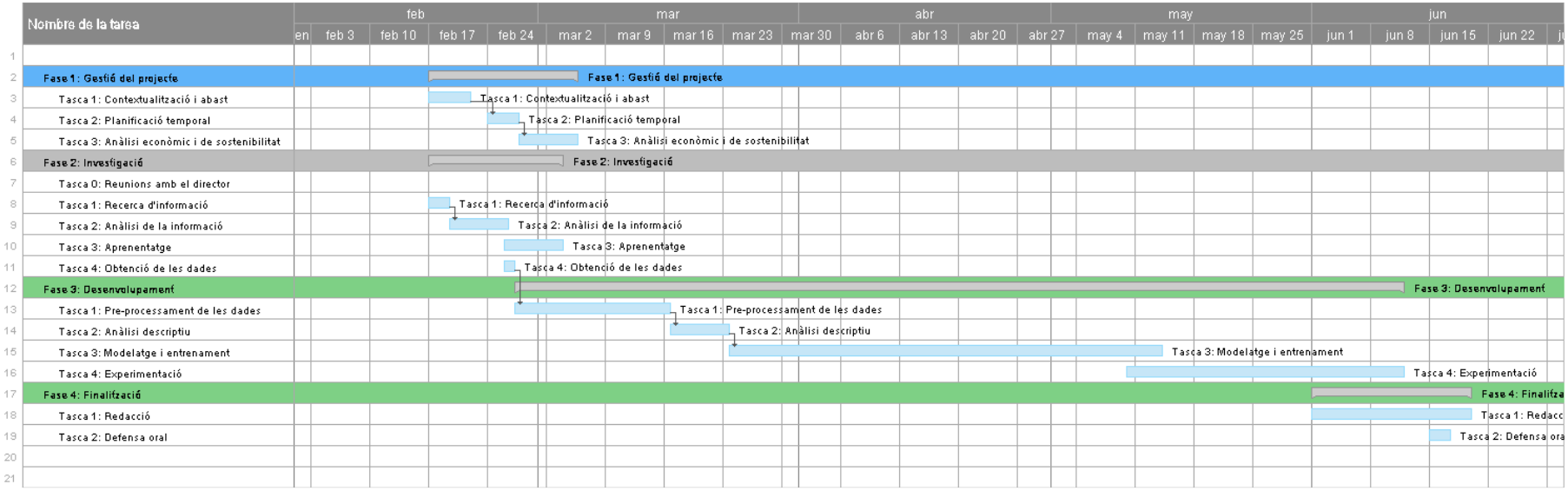
Finalment, tenim l'elaboració de la presentació del treball i la seva defensa davant d'un tribunal.

6.2 - Diagrama de Gantt

Com s'ha comentat anteriorment, les tasques 3 i 4 de la fase 3, tot i tenir un nombre elevat d'hores consistiran en diferents esprints d'una durada inferior al total d'hores. Per la tasca 3, cadascun d'aquests esprints consistirà en la creació d'un model a partir d'un algorisme concret, l'entrenament del model amb aquell algorisme i el testeig d'error i precisió d'aquell algorisme. Aquest procés es repetirà amb diferents algorismes, per tant, la tasca 3 consistirà en diferents esprints que juntament sumaran les 150 hores totals.

Similar serà la tasca 4 de la fase 3, on realitzarem la validació de cadascun dels models que haguem trobat.

Nom de la tasca	Data d'inici	Data final	Duració	Dependències
Fase 1: Gestió del projecte	**17/02/20**	**05/03/20**		
Tasca 1: Contextualització i abast	17/02/20	21/02/20	20h	
Tasca 2: Planificació temporal	24/02/20	27/02/20	15h	F1T1
Tasca 3: Anàlisi econòmica i sostenibilitat	27/02/20	05/03/20	20h	F1T2
Fase 2: Investigació	**17/02/20**	**03/03/20**		
Tasca 0: Reunions amb el director			5h	
Tasca 1: Recerca d'informació	17/02/20	19/02/20	10h	
Tasca 2: Anàlisi de la informació	19/02/20	26/02/20	20h	F2T1
Tasca 3: Aprenentatge	26/02/20	03/03/20	20h	
Tasca 4: Obtenció de les dades	26/02/20	27/02/20	5h	
Fase 3: Desenvolupament	**27/02/20**	**01/06/20**		
Tasca 0: Reunions amb el director			5 h	
Tasca 1: Pre-processament de les dades	27/02/20	16/03/20	50h	F2T4
Tasca 2: Anàlisi descriptiva	16/03/20	23/03/20	20h	F3T1
Tasca 3: Modelatge i entrenament	23/03/20	14/05/20	150h	F3T2
Tasca 4: Validació	10/04/20	01/06/20	100h	
Fase 4: Finalització	**01/06/20**	**19/06/20**		
Tasca 1: Redacció	01/06/20	19/06/20	60h	
Tasca 2: Defensa oral	15/06/20	17/06/20	10h	


Nombre de la tarea
feb
mar
abr
may
jun
en
feb 3
feb 10
feb 17
feb 24
mar 2
mar 9
mar 16
mar 23
mar 30
abr 6
abr 13
abr 20
abr 27
may 4
may 11
may 18
may 25
jun 1
jun 8
jun 15
jun 22
1
2 Fase 1: Gestió del projecte
3 Tasca 1: Contextualització i abast
4 Tasca 2: Planificació temporal
5 Tasca 3: Anàlisi econòmic i de sostenibilitat
6 Fase 2: Investigació
7 Tasca 0: Reunions amb el director
8 Tasca 1: Recerca d'informació
9 Tasca 2: Anàlisi de la informació
10 Tasca 3: Aprenentatge
11 Tasca 4: Obtenció de les dades
12 Fase 3: Desenvolupament
13 Tasca 1: Pre-processament de les dades
14 Tasca 2: Anàlisi descriptiu
15 Tasca 3: Modelatge i entrenament
16 Tasca 4: Experimentació
17 Fase 4: Finalització
18 Tasca 1: Redacció
19 Tasca 2: Defensa oral
20
21

6.3 - Canvis en la planificació

Un cop arribat a la fita de seguiment hi ha hagut diversos canvis respecte la planificació inicial. El major canvi ha estat que, com es va predir que podia passar en la gestió de riscos en el pla de treball inicial, hi ha hagut un error en la planificació temporal de les tasques de Modelatge i Entrenament. Això ha provocat que s'hagi de reduir el temps dedicat a la tasca de Validació, dedicant aquelles hores a la tasca de Modelatge i Entrenament. A continuació explicarem els motius que han provocat l'error en la predicció dels temps inicial i dels canvis en la planificació inicial.

En la gestió de risc es va plantejar que el temps d'entrenament, modelat i validació necessités més temps del plantejat, cosa que ha acabat passant. Es va argumentar que s'intentaria paral·lelitzar el codi per millorar el rendiment i executar amb subconjunt de dades per trobar els millors paràmetres, per finalment utilitzar totes les dades per entrenar el model final. S'han realitzat les accions anteriors, i s'ha aconseguit una gran millora en els temps d'execució gràcies a la paral·lelització, tot i això els temps d'execució dels diferents algoritmes ha estat superior al estimat inicialment, arribant a ser de 4 a 8 hores en alguns casos. En molts casos, degut a la paral·lelització del codi, provoca que l'ordinador estigui a més del 90% d'ús en CPU i Memòria, provocant que durant el temps d'execució dels algoritmes no es pogués utilitzar l'ordinador per altres tasques. Per tant, tot i utilitzar diferents eines per intentar reduir l'impacte d'aquest problema, els temps d'execució dels diferents algoritmes han estat superiors als esperats i han provocat un augment del temps dedicat a la tasca de Modelatge i Entrenament.

Un altre problema que ha provocat el canvi en la planificació inicial ha estat els resultats obtinguts inicialment. Abans de començar la tasca de modelatge es va realitzar una consulta amb el tutor per plantejar quins serien els millors algoritmes a utilitzar. Després de provar diferents algoritmes d'aprenentatge automàtic i diferents paràmetres per cadascun dels algoritmes, es van aconseguir precisions dels models molt baixes, aproximadament totes elles del 60-70%. Donats els pobres resultats obtinguts, es va realitzar una altre reunió amb el tutor per avaluar alternatives i es va decidir utilitzar *ensembled classifiers* per tal d'intentar trobar un model amb millor precisió. Això ha provocat que s'hagi hagut de dedicar temps extra en entrenament de nous models, els quals al ser més complexos que els anteriors també requereixen de més temps, per acabar aconseguint un model amb una precisió del 99%.

En conclusió, s'ha hagut de reduir el temps de la tasca de Validació i dedicar aquelles hores a la tasca de Modelatge i Entrenament degut a una mala planificació en el temps necessari, i sobretot als mals resultats obtinguts inicialment. Això no ha provocat un gran impacte en el la planificació temporal treball ja que, tot i reduir les hores dedicades a la validació dels models, aquestes s'han utilitzat per realitzar una validació més exhaustiva en aquells models que presenten inicialment unes millors precisions. Per tant, la planificació actual seria la presentada a la següent taula.

Nom de la tasca	Data d'inici	Data final	Duració	Dependències
Fase 1: Gestió del projecte	**17/02/20**	**05/03/20**		
Tasca 1: Contextualització i abast	17/02/20	21/02/20	20h	
Tasca 2: Planificació temporal	24/02/20	27/02/20	15h	F1T1
Tasca 3: Anàlisi econòmica i sostenibilitat	27/02/20	05/03/20	20h	F1T2
Fase 2: Investigació	**17/02/20**	**03/03/20**		
Tasca 0: Reunions amb el director			5h	
Tasca 1: Recerca d'informació	17/02/20	19/02/20	10h	
Tasca 2: Anàlisi de la informació	19/02/20	26/02/20	20h	F2T1
Tasca 3: Aprenentatge	26/02/20	03/03/20	20h	
Tasca 4: Obtenció de les dades	26/02/20	27/02/20	5h	
Fase 3: Desenvolupament	**27/02/20**	**11/06/20**		
Tasca 0: Reunions amb el director			5 h	
Tasca 1: Pre-processament de les dades	27/02/20	16/03/20	50h	F2T4
Tasca 2: Anàlisi descriptiva	16/03/20	23/03/20	20h	F3T1
Tasca 3: Modelatge i entrenament	23/03/20	01/06/20	200h	F3T2
Tasca 4: Validació	10/04/20	01/06/20	75h	
Fase 4: Finalització	**01/06/20**	**19/06/20**		
Tasca 1: Redacció	01/06/20	19/06/20	60h	
Tasca 2: Defensa oral	15/06/20	17/06/20	10h	

7 - Anàlisi econòmica i de sostenibilitat

7.1 - Recursos

7.1.1 - Recursos humans

- Estudiant: el principal recurs del projecte ja que és l'encarregat de gestionar i desenvolupar el projecte.
- Director/Tutor: L'encarregar de guiar i ajudar al estudiant durant el desenvolupament del projecte.

7.1.2 - Recursos materials

- Material d'oficina: taula, cadira, papers, etc, tot el material d'oficina per la realització del projecte.
- Hardware:
    - Ordinador: la principal eina de treball
    - Perifèrics: pantalles, teclat, ratolí, etc.
- Software:
    - Gitlab: per el control de versions del projecte.
    - Outlook: pel contacte amb el tutor.
    - R-Studio: per desenvolupar les parts del projecte que es realitzaran en R.
    - Atom: editor de text per programar en Python.
    - Bash Ubuntu: per executar els scripts de Python.
    - Office: per la redacció de la documentació

7.1.3 - Despeses generals

Les diferents despeses generals (aigua, llum, electricitat, etc) a utilitzar per la realització del projecte.

7.2 - Pressupost

7.2.1 - Costos en recursos humans

En aquest projecte, com hem comentat anteriorment, hi intervenen 2 persones: el director i l'estudiant. El director tindrà el rol de "Cap de projecte" i l'estudiant d'"Analista programador", "Programador" i "Analista", segons les funcions i salaris trobats a internet [37]. Pel càlcul del Salari/Hora es tindrà en compte una jornada laboral de 8 hores diàries i que el 2020 té 253 dies laborables [38].

Taula 4: Salaris segons rol

Rol	**Salari Anual**	**Salari/Hora**
Cap de projecte	30.700 €	15.27 €
Analista programador	30.300 €	14.97 €
Programador	29.800 €	14.72 €
Analista	25.000 €	12.35

Taula 5: Assignació de tasques segons rol

Nom de la tasca	**Assignació**
Fase 1: Gestió del projecte	Cap de projecte
Tasca 1: Contextualització i abast	Cap de projecte
Tasca 2: Planificació temporal	Cap de projecte
Tasca 3: Anàlisi econòmica i de sostenibilitat	Cap de projecte
Fase 2: Investigació	
Tasca 0: Reunions amb el director	Cap de projecte
Tasca 1: Recerca d'informació	Analista
Tasca 2: Anàlisi de la informació	Analista
Tasca 3: Aprenentatge	Analista
Tasca 4: Obtenció de les dades	Programador
Fase 3: Desenvolupament	
Tasca 1: Pre-processament de les dades	Programador
Tasca 2: Anàlisi descriptiu	Programador
Tasca 3: Modelatge i entrenament	Analista programador
Tasca 4: Experimentació	Analista programador
Fase 4: Finalització	Analista programador
Tasca 1: Redacció	Analista programador
Tasca 2: Defensa oral	Analista programador

Es repartiran les tasques esmentades en l'anterior entrega entre aquests dos rols per fer el càlcul dels costos humans. Al cap del projecte se li assignen les tasques de la fase 1, és a dir, les de gestió del projecte i la tasca dedicada a reunions amb el director. Al analista se li assignen les tasques d'anàlisi de la informació. Al programador les tasques inicials del desenvolupament del projecte. Al analista programador se li assignen la resta de les tasques. A continuació tenim un resum del total d'hores de cada rol i els costos humans final.

Taula 6: Costos humans

Rol	**Hores**	**Cost**
Cap de projecte	65 h	992,55 €
Analista programador	320 h	4790,4 €
Programador	50 h	736 €
Analista	75 h	926,25 €
	Total	7445,2 €

7.2.2 - Costos en recursos materials

Abans de mostrar la taula amb els diferents costos materials es farà unes breus consideracions que s'han tingut en compte en la realització dels càlculs.

- Hardware: entendrem que s'amortitza quan l'antiguitat és superior als 3 anys. Tot i això, el material a utilitzar en aquest projecte (ordinador i perifèrics) tenen una antiguitat inferior als 3 anys, per tant no es poden amortitzar.

- Software: Tot el software que s'utilitzarà pel desenvolupament d'aquest projecte (R-Studio, Gitlab, Python, Atom, etc) és "open-source" i per tant, completament gratuït.

- Material d'oficina: en aquest cas sí que està amortitzat eines bàsiques a utilitzar (taula i cadira). Per tant, només es comptarà un extra de 20€ de possibles despeses en papers, bolígraf, etc.

- Preu/Hora: S'ha tingut en compte que hi ha 253 dies laborals en una jornada de 8 hores, i que s'amortitza en 3 anys.

- Hores projecte: s'entén que tot el projecte es realitzarà utilitzant els mateixos recursos materials (ordinador i perifèrics) i per tant, se'ls hi ha adjudicat la totalitat d'hores del projecte, exceptuant les hores dedicades a reunions amb el director.

Taula 7: Costos en recursos materials

Recurs	Cost total	Cost / hora	Hores projecte	Cost projecte
Ordinador	1200 €	0.19 €	500 h	104,5 €
Perifèrics	250 €	0.04 €	500 h	25 €
Software	0 €	0 €	500 h	0 €
Material oficina	20 €	-	-	20€
			Total	149,50 €

7.2.3 - Costos en despeses generals

A continuació tenim una taula dels costos en despeses generals, de la qual hem de tenir en compte les següents consideracions:

- Consum elèctric: L'ordinador consumeix aproximadament uns 180W/h i la pantalla 40W/h. La resta de perifèrics van connectats a l'ordinador i per tant el seu consum elèctric està inclòs dins del de l'ordinador. El preu aproximat del kWh a Espanya és de 0,1127930 €/kWh [39].
- Internet: Tindrem en compte una tarifa de 100Mb amb un preu aproximat de 25€ mensuals.
- Temps: Al igual que en els costos de recursos materials s'entén que el consum elèctric serà constant durant les 500 hores de duració del projecte ja que totes les tasques es realitzaran en ordinador, i per tant, hi haurà un consum elèctric constant.

Taula 8: Costos en despeses generals

Despesa	Temps	Consum total	Cost
Consum elèctric	500 h	121 kW	13,65 €
Internet	6 mesos	-	150 €
		Total	163,65 €

7.2.4 - Impostos

Finalment tenim els diferents impostos a pagar. Pels recursos humans s'aplicarà el cost de la seguretat social, mentre que per els materials i les despeses generals s'aplica l'IVA.

Taula 9: Impostos

Impost	Percentatge	Cost	Total
Seguretat social	23.60 %	7542,95 €	1780,14 €
IVA	21 %	313,15 €	65,76 €
		Total	1845,9 €

7.2.5 - Resum

Per finalitzar el pressupost d'aquest projecte presentem la següent taula com a resum dels diferents costos que s'han analitzat en els apartats anteriors.

Taula 10: Pressupost final

Concepte	**Cost**
Recursos humans	7445,2 €
Recursos materials	149,50 €
Despeses generals	163,65 €
Impostos	1845,9 €
Total	9702 €

7.2.6 - Control de gestió

Pel control dels costos de recursos humans s'utilitzarà un registre de les hores dedicades a les diferents tasques i quin dels rols la duu a terme. D'aquesta manera es podrà seguir si s'estan complint les estimacions realitzades en el diagrama de Gantt i per tant, si s'està controlant el pressupost.

Com hem explicat en la gestió de riscos, donat que aquest projecte és un treball d'investigació, si acabem una tasca en menys hores de les plantejades s'aprofiten aquelles hores en altres tasques, especialment en les d'experimentació i redacció. Per altre banda, si hi ha tasques amb una duració superior a la prevista, s'utilitzaran menys hores en altres tasques (especialment en modelatge i experimentació). Per tant, respecte als recursos humans, donat el temps limitat que es té, es complirà el pressupost ja que s'hi hauran de dedicar el total d'hores plantejades.

Respecte els recursos materials la única desviació possible del pressupost seria que avariessin l'ordinador o algun dels perifèrics. Donat que són bastant nous, és difícil que es doni aquesta situació. Però en cas que passés, es disposa d'un ordinador antic (el qual al tenir una antiguitat superior als 4 anys ja està amortitzat) el qual es podria fer servir en cas d'avaria de l'ordinador actual. Per tant, el cost final dels recursos materials no hauria de variar en el transcurs del projecte.

7.3 - Informe de sostenibilitat

7.3.1 - Dimensió econòmica

Anteriorment ja hem vist que s'ha fet una gestió de possibles riscos que tinguessin un impacte econòmic en el projecte, a la vegada que s'han quantificat el cost dels recursos humans i materials utilitzats. Com els recursos humans és l'estudiant realitzant el projecte, i els materials un ordinador que no s'ha espatllat, no hi ha hagut ningun increment en els costos del projecte. A la vegada, és impossible reduir els costos ja que, més enllà del consum d'electricitat, s'han utilitzat el mínim de recursos indispensables per la realització del projecte.

A la vegada, al ser un projecte d'investigació no hi haurà costos associats durant la vida útil del projecte ja que, un cop publicat el treball, els costos durant la vida útil del projecte seran 0. Qualsevol tipus d'actualització en el projecte seria segurament un projecte per separat ja que seria per fer una investigació diferent a la realitzada en aquest projecte. Per tant, al ser un projecte que no té una finalitat lucrativa, no hi hauria escenaris que afectessin la viabilitat econòmica del projecte. L'única situació possible que afectés la viabilitat econòmica era que s'espatllés l'ordinador durant la realització del treball (cosa que no va succeir), i tot i així es tenien alternatives a cost 0.

7.3.2 - Dimensió social

El desenvolupament del projecte m'ha ajudat a fer veure que l'estudi de dades, i per tant tota l'àrea de aprenentatge automàtic, mineria de dades, etc, té marge per créixer en l'àmbit professional relacionat amb l'esport. Al aplicar-se a una àrea com el bàsquet, a la qual li he dedicat una gran part de la meva vida, ha servit per reflexionar de quins són els diferents factors importants en la realització d'un tir, i veure que no només depèn del talent d'un jugador.

Al haver aconseguit crear un model amb una alta precisió s'ha aconseguit veure que hi ha molts factors que afecten a que un tir entri o falli, i que es pot predir sense tindre en compte l'habilitat del jugador que tira. I al ser un projecte d'investigació sobre el tir en el bàsquet, no tindrà cap impacte perjudicial o de dependència en les persones afectades, en aquest cas els jugadors.

7.3.3 - Dimensió ambiental

És difícil mesurar l'impacte ambiental d'un projecte d'investigació que l'únic recurs material que necessitava és un ordinador. L'impacte ambiental d'aquest projecte és l'electricitat utilitzada per l'ordinador, pantalles, etc, i la llum de l'habitació on està situat. Per tant, totes les accions que es poden dur a terme per reduir l'impacte ambiental serien utilitzar el màxim possible la llum de l'exterior o utilitzar bombetes de baix consum i apagar l'ordinador en els moments que no s'estava fent servir. S'han seguit aquestes accions durant la realització del projecte, per tant, seria bastant difícil reduir els recursos utilitzats durant el desenvolupament. Al haver-se realitzat el projecte durant una època on no fa molta calor ni molt de fred, no hi ha hagut un consum d'electricitat o gas associat a calefaccions o aires. En cas d'utilitzar-se seria igual que amb la llum, utilitzar-los només en els casos necessaris i havent buscat alternatives (posar-se un jersei o obrir la finestra).

A la vegada, al ser un treball d'investigació l'impacte que pot tindre en la petjada ecològica és nul. Al ser un sistema de predicció de tir en la NBA, la utilització d'aquest treball per intentar millorar el rendiment dels jugadors no implicaria ningun canvi important que pogués millorar o empitjorar la petjada ecològica.

8 - Conclusions

En l'entorn esportiu, al igual que en moltes altres àrees, està augmentant la quantitat de dades que són recollides i analitzades per intentar millorar el rendiment. En aquest projecte ens hem centrat en un esport com és el bàsquet. Més concretament en la lliga americana (NBA), la qual al ser la més important a nivell mundial també és la que disposa d'un major volum de dades. L'objectiu del treball ha estat desenvolupar un sistema de predicció de tir a partir de dades relacionades amb el moment del tir, com per exemple la distància a cistella, el temps restant o el tipus de tir, entre altres. Hem treballat amb més de 200.000 observacions, és a dir, tirs diferents en una temporada completa de la NBA.

Per assolir el nostre objectiu hem estudiat i treballat amb diferents algoritmes d'aprenentatge automàtic i especialment en conjunts de classificadors (*ensemble of classifiers)*. Hem provat una gran varietat de possibles models, des de xarxes neuronals artificials fins a realitzar un *stacking* utilitzant fins a 6 models diferents. Finalment hem aconseguit el nostre objectiu ja que hem obtingut un model amb una precisió del 99%, és a dir, un model que és capaç de predir si un tir és encertat o fallat amb menys d'un 1% d'error. Per assolir aquest objectiu també s'han aconseguit els diferents sub-objectius ja que, per aconseguir aquest model final s'ha hagut d'identificar el millor model possible pel nostre problema i validar correctament els nostres models per minimitzar els problemes d'ajust. A la vegada, durant el tractament de les dades i la seva preparació per l'entrenament dels models, hem creat una variable que identifica diverses zones d'un camp de bàsquet, la qual posteriorment hem vist que és relativament important, al igual que hem identificat altres factors importants com la posició, el temps restant o la distància del defensor.

8.1 - Treball futur

I quines serien les posteriors investigacions o millores que es poden realitzar? En aquest apartat comentarem unes quantes. Ens centrarem en aquelles d'àmbit esportiu relacionades amb el bàsquet i a nivell d'informació que es pot recollir en un partit. No comentarem, per exemple, models per predir qui serà campió de la lliga, qui serà el jugador més valuós o altres tipus de prediccions que actualment les realitzen diversos mitjans de comunicació i que utilitzen dades amb un component més subjectiu.

Una millora que es podria fer seria treballar amb dades més actuals. En aquest projecte, degut a canvis en la recollida de dades per part la NBA, s'ha treballat amb dades de la temporada 2014-15. Per tant, si la lliga tornés a recollir la mateixa informació que anteriorment, seria interessant realitzar una investigació similar amb dades actuals. Un altre enfocament seria treballar amb dades de diferents anys i fer una comparació temporal ja que, per exemple es podria incloure informació de la temporada en la que s'està i veure si les característiques del tir ha evolucionat amb el pas del temps i quin efecte té.

Un altre millora diferent seria afegir informació sobre l'habilitat del jugador que tira. En aquest projecte l'únic que s'ha inclòs ha estat el nombre de tirs realitzats pel jugador i el seu percentatge de tir. En aquest cas el problema vindria en com mesurar l'habilitat de tir d'un jugador ja que és més complicat del que pot semblar. S'hauria de tindre en compte la seva habilitat segons la posició o zona des de on s'efectua el tir, el temps restant, temps que porta jugat el jugador (per mesurar el cansansament), si està en una ratxa de tirs encertats/fallats, etc. En conjunt, s'hauria d'incloure molta informació per aconseguir mesurar correctament i amb precisió l'habilitat de tir d'un jugador per un tir concret.

Relacionat amb l'anterior punt, també seria interessant fer l'anàlisi per un jugador concret. En aquest cas, intentant mesurar el millor possible la seva habilitat en el tir, es podria aconseguir un model que podria ajudar al jugador en concret a millorar el seu joc, veure en quines situacions falla més, o quines serien les situacions que el model prediu com a més favorables per encertar.

I no hauria de perquè centrar-se en el tir una futura investigació en l'àmbit del bàsquet. Per exemple es poden estudiar factors en un partit que fan que un equip guanyi. Al igual que tenim dades per tots els tirs realitzats en un partit, també tenim informació de tota la resta d'accions que ocorren. Per tant, es podrien analitzar totes les diferents jugades d'un partit pels dos equips i veure quin efecte tenen en el resultat final. En aquest cas, més que un problema de classificació seria millor si fos de regressió i veiem com diverses accions fan augmentar o disminuir les probabilitats de victòria d'un equip o el seu rival.

En conclusió, hi ha moltes possibles futures investigacions relacionades amb les temàtica vista en aquest projecte d'investigació. El món de l'esport està començant a recollir un gran volum de dades i és important treballar amb elles i estudiar-les per intentar millorar el rendiment.

9 - Bibliografia

[1] FIB "Facultat d'Informàtica de Barcelona" [en línia]. [Consulta: 19/02/2020] https://www.fib.upc.edu/

[2] UPC "Universitat Politècnica de Catalunya" [en línia]. [Consulta: 19/02/2020] https://www.upc.edu/

[3] "National Basketball Association" A Wikipedia [en línia]. [Consulta: 19/02/2020]

https://en.wikipedia.org/wiki/National_Basketball_Association

[4] "Apredizaje Automatico" A Wikipedia [en línia]. [Consulta: 21/02/2020]

https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico

[5] IBÁÑEZ, S. J., et al. La eficacia del lanzamiento a canasta en la NBA: Análisis multifactorial.(Shot efficacy in the NBA: A multifactorial analysis). *Cultura_Ciencia_Deporte*, 2009, vol. 4, no 10, p. 39-47.

[6] CIAMPOLINI, Vitor, et al. Factors associated with basketball field goals made in the 2014 NBA finals. *Motriz: Revista de Educação Física*, 2017, vol. 23, no 4.

[7] VELÁZQUEZ CARRICONDO, Juan Antonio. Técnicas de minería de datos aplicadas en baloncesto. 2018.

[8] PIETTE, James; ANAND, Sathyanarayan; ZHANG, Kai. Scoring and shooting abilities of NBA players. *Journal of Quantitative Analysis in Sports*, 2010, vol. 6, no 1.

[9] REICH, Brian J., et al. A spatial analysis of basketball shot chart data. *The American Statistician*, 2006, vol. 60, no 1, p. 3-12.

[10] GARCÍA, Javier, et al. Estudio de la gestoforma del lanzamiento a canasta en la liga EBA. *Retos. Nuevas tendencias en Educación Física, Deporte y Recreación*, 2008, no 14, p. 17-21.

[11] IZAURIETA, Fernando; SAAVEDRA, Carlos. Redes neuronales artificiales. *Departamento de Física, Universidad de Concepción Chile*, 2000.

[12] SALAS, Rodrigo. Redes neuronales artificiales. *Universidad de Valparaıso. Departamento de Computación*, 2004, vol. 1.

[13] "Aprendizaje basado en arboles de decisión" A Wikipedia [en línia] [Consultat:06/06/2020] https://es.wikipedia.org/wiki/Aprendizaje_basado_en_%C3%A1rboles_de_decisi%C3%B3n

[14] "C4.5" A Wikipedia [en línia] [Consultat:06/06/2020] https://es.wikipedia.org/wiki/C4.5

[15] LOPEZ TAKEYAS, Bruno. "Algoritmo C4.5" *Instituto Tecnologico Nuevo Laredo* [en línia] [Consultat:06/06/2020] http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/C4.5/C4.5(2005-II-B).pdf

[16] GANDHI, Rohith. "Support Vector Machine - Introduction to Machine Learning Algorithms" [en línia] [Consultat:11/06/2020] https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[17] AMAT RODRIGO, Joaquin. "Máquinas de Vector Soporte". [en línia] [Consultat:11/06/2020] https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines

[18] Aprende machine learning "Classificar con K-Nearest-Neighbor" [en línia] [Consultat:11/06/2020] https://www.aprendemachinelearning.com/clasificar-con-k-nearest-neighbor-ejemplo-en-python/

[19] "Random Forest" A Wikipedia [en línia] [Consultat:11/06/2020] https://en.wikipedia.org/wiki/Random_forest

[20] D'SOUZA, Jocelyn. "A quick guide to boosting in ML" [en línia] [Consultat:11/06/2020] https://medium.com/greyatom/a-quick-guide-to-boosting-in-ml-acf7c1585cb5

[21] Rulequest "C5.0: An informal tutorial" [en línia] [Consultat:11/06/2020] https://www.rulequest.com/see5-unix.html

[22] GeeksforGeeks "Stacking in Machine Learning" [en línia] [Consultat:11/06/2020] https://www.geeksforgeeks.org/stacking-in-machine-learning/

[23] "Validación cruzada". A Wikipedia [en línia] [Consultat:11/06/2020] https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

[24] Scrum "Metodologia àgil de treball" [en línia]. [Consulta: 22/02/2020] https://www.scrum.org/

[25] GitLab " Servei de control de versions i gestor de repositoris" [en línia]. [Consulta: 22/02/2020] https://about.gitlab.com/

[26] Taiga "Sistema de gestió de projectes" [en línia]. [Consulta: 22/02/2020] https://taiga.io/

[27] "Validación cruzada" A Wikipedia [en línia]. [Consulta: 22/02/2020] https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

[28] NBA "NBA Advanced Stats" [en línia] [Consulta: 23/05/2020] https://stats.nba.com/

[29] NBAsavant "Shot Tracker" [en línia] [Consulta: 23/05/2020] https://nbasavant.com/shot_search.php

[30] "Feet to meters" Metric Conversions [en línia] [Consultat: 06/06/2020] https://www.metric-conversions.org/length/feet-to-meters.htm

[31] "Temporada 2014-14 de la NBA" A Wikipedia. [en línia] [Consultat: 06/06/2020] https://es.wikipedia.org/wiki/Temporada_2014-15_de_la_NBA

[32] SHARMA, Mohit "Functions and packages for feature selection in R" [en línia] [Consultat:06/06/2020]

https://datasciencebeginners.com/2018/11/26/functions-and-packages-for-feature-selection-in-r/#Example_2_Recursive_Feature_Elimination_Method

[33] STIGLIC, Gregor, et al. Comprehensive decision tree models in bioinformatics. *PloS one*, 2012, vol. 7, no 3.

[34] Learn by Marketing "Decision Trees in R" [en línia] [Consultat:11/06/2020] http://www.learnbymarketing.com/tutorials/rpart-decision-trees-in-r/

[35] Guia Treball de Fi Grau per a estudiants d'informàtica. [en línia]. [Consulta: 26/02/2020] https://www.fib.upc.edu/ca/estudis/graus/grau-en-enginyeria-informatica/treball-de-fi-de-grau

[36] "European Credit Transfer and Accumulation System" A Wikipedia. [en línia]. [Consulta: 26/02/2020] https://es.wikipedia.org/wiki/European_Credit_Transfer_and_Accumulation_System

[37] "Informe Empleo Informática - Marzo 2020". [en línia]. [Consulta: 05/03/2020] https://www.tecnoempleo.com/informe-empleo-informatica.php

[38] "Días laborables" [en línia]. [Consulta: 05/03/2020] https://www.dias-laborables.es/dias_laborables_feriados_2020.htm

[39]"Tarifas gas luz" Precio del kWh [en línia] [Consulta: 06/03/2020] https://tarifasgasluz.com/faq/precio-kwh

[40] QUINLAN, J. Ross. *C4. 5: programs for machine learning*. Elsevier, 2014.