UNIVERSITAT POLITÈCNICA DE CATALUNYA
GRAU EN CIÈNCIA I ENGINYERIA DE DADES

Predicció sobre la selecció de l'All Star de la NBA

Adrián Cerezuela Hernández

Joan Ot Vidal Camps

Assignatura: **Aprenentatge Automàtic 1**

Data d'entrega: **9 de juny de 2023**

Juny de 2023

Índex

1 INTRODUCCIÓ

Aquest és un projecte avaluador per l'assignatura d'Aprenentatge Automàtic 1, del grau en Ciència i Enginyeria de Dades. L'objectiu d'aquest és predir quins jugadors seran seleccionats per l'All Star Game de l'NBA a partir de les seves estadístiques durant la temporada. A més, tenint en compte que el dataset seleccionat consta de tres temporades, un altre objectiu és poder trobar si el model és estable, és a dir, si no té canvis importants, o si diferentment, canvia considerablement entre temporades.

La motivació principal per escollir sobre quin àmbit realitzar el nostre projecte resideix bàsicament en el nostre interés per l'esport, en concret pel bàsquet. Per tant, tenir l'opció d'aplicar les tècniques vistes a classe per modelar aquest fet cridava la nostra atenció. Així mateix, no hem hagut de realitzar estudi previ sobre el tema, ja que el nostre coneixement sobre el mateix és molt ampli ja de base.

1.1 Descripció de les dades

Per desenvolupar aquest projecte, hem utilitzat el dataset *NBA Players 2016-2019*, d'OpenML. Com ja s'ha esmentat, aquest consta de dades individuals sobre cada jugador al llarg de les temporades 2016-2017, 2017-2018 i 2018-2019. Inicialment, conté 1408 instàncies i 45 atributs. Del total de jugadors que cobreixen les instàncies, només 73 son seleccionats. És habitual que un jugador repeteixi participació en anys consecutius, de manera que seran molts menys portant això al context real. Normalment, els seleccionats totals per temporada són 24, 12 per conferència. No obstant, no és un nombre fix, ja que poden haver lesions i, per tant, substitucions. Quan això passa el nombre és major que 24, ja que compta la selecció tant pel jugador incial com pel seu substitut.

Al dividir les dades per temporades, trobem que tenim 436, 498 i 474 instàncies, respectivament. A més de voler comprobar l'estabilitat del model entre temporades, una altra raó per realitzar aquesta partició és eliminar la component temporal inherent al dataset, ja que podria portar-nos inconvenients a l'hora de modelar.

2 TRACTAMENT DE LES DADES

2.1 Preprocessat

Previ a la divisió en tres parts, realitzarem el tractament de tot el conjunt de dades, dividint aquesta tasca en les següents parts: Selecció i extracció de features, Codificació d'atributs, Tractament de missing values i outliers i Normalització de les dades.

En quant als atributs de les nostres dades, trobem que alguns d'ells no són rellevants a l'hora de portar a terme el nostre anàlisi i, per tant, han de ser eliminats per tal de no tenir variables adicionals que no aportin res. Aquests són, concretament, *Pos2*, *Rk*, *Player.x* i *Player.ID*. A més, per comoditat d'ús, cal fer un reanomenament dels atributs que representen els percentatges de llançament, passant d'expressar-los com "*XXX.*" a "*XXX_Per*".

Després de seleccionar i extreure els atributs, cal tractar els missing values. Inicialment, en tenim als percentatges de llançament, als vots i rankings dels jugadors i la premsa, al salari i a les visites mitjanes a la seva pàgina de Wikipèdia.

- Primerament, considerarem que si no tenim dades sobre percentatges d'un jugador és per què no ha realitzat cap llançament corresponent a la puntuació en qüestió i, per tant, aquests valors seran imputats a 0.

- Els atributs de vots i rankings contenen valors de l'1 al 10, i imputar aquests missing values per nearest neighbours, tal com farem amb el salari o les visites, podria assignar un "rank" alt a algun jugador que realment no hagi obtingut cap puntuació. Per aquesta raó, seran imputats a 0. Tot i això, el fet de posar a 0 el "rank" pot causar problemes més endavant, ja que en principi com més baix el "rank", millor. Per aquest motiu, tindrem present aquest fet i tornarem enrere si veiem que aquesta decisió no era la més indicada.

- En quant al salari i les *mean views*, utilitzarem el mètode de k-NearestNeighbours per imputar aquests valors, amb k=2. Aquesta estimació es basa en la similaritat de les característiques entre l'observació amb missing values i els seus k veïns més propers.

Per a que els algorismes de machine learning puguin processar les variables categòriques, utilitzarem One Hot Encoding per convertir variables categòriques en una representació numèrica, creant columnes addicionals binàries per a cada categoria, on s'assigna un valor de 1 si pertany a la categoria i 0 si no ho fa. També la nostra variable resposta passarà per aquest procés, convertint els 'Yes' i 'No' en 0 si no és seleccionat i 1 si ho és.

En quant als outliers, ens pot interessar eliminar els que es corresponen amb els percentatges de llançament. La raó principal és que si el valor és molt alt o, per contra, molt baix, pot ser degut a que aquest jugador no hagi realitzar suficients llançaments a cistella i, per tant, pot no ser un indicador de la qualitat real del jugador en qüestió. Cal esmentar que a l'atribuir un percentatge de 0 als valors nuls, no eliminarem els outliers baixos de cap dels percentatges.

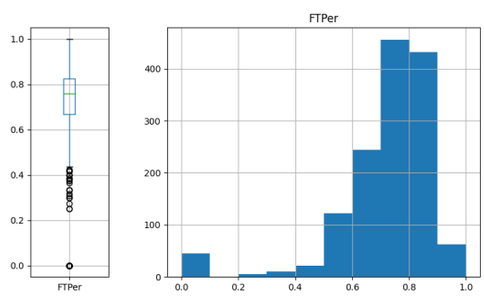

Figura 1: Histograma de *FTPer*

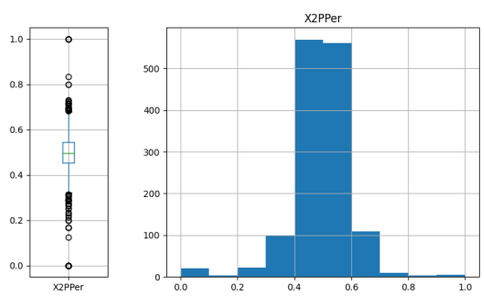

Figura 2: Histograma de *X2PPer*

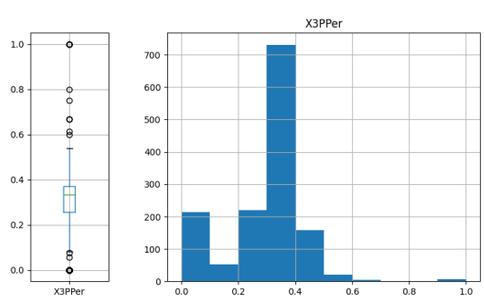

Figura 3: Histograma de *X3PPer*

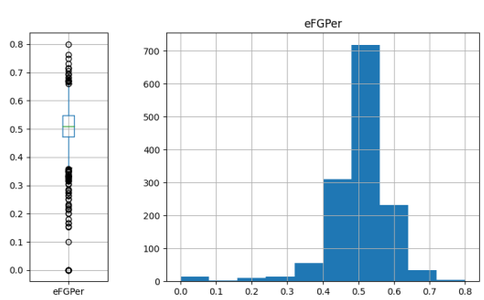

Figura 4: Histograma de *eFGPer*

Fixant-nos en les Figures (2) i (3), podem observar bastants outliers, dels quals eliminarem només els valors iguals a 1 en aquestes variables, és a dir a un 100% d'efectivitat. La raó principal té relació amb el que s'ha esmentat abans, pot ser que no s'hagin produit suficients llançaments per a que el percentatge representi un valor traslladable a un context realista. En vista de no poder decidir el

valor màxim possible, la resta de valors no es veuran modificats. Un cop eliminats aquests valors, no hauriem de trobar valors atípics a l'*eFGPer*, ja que aquest percentatge surt del nombre de llançaments anotats més la meitat dels triples anotats, entre el total de llançaments intentats [Bas]. En quant a la Figura (1), no trobem cap valor anòmal ni massa elevat.

Com a últim pas del preprocessat, normalitzar les dades és un pas important del preprocessat. Implica escalar els valors del dataset a un rang estàndard entre 0 i 1, per tal de garantir que es modela amb valors que estan en una escala similar, evitant la problemàtica de que el model pugui captar només valors molt llunyans d'aquesta i fer unes prediccions errònies. D'entre les transformacions més comuns, nosaltres hem utilitzat el Min-Max Scaling:

$$\frac{X - X_{min}}{X_{max} - X_{min}}$$

2.2 Visualització de la informació

A l'hora de visualitzar la informació ens interessen tres aspectes generals: la correlació entre les nostres variables, la distribució d'aquestes un cop normalitzades, i com es distribueix la nostra variable resposta en relació a certs atributs.

Amb l'objectiu de representar les correlacions entre les variables, desenvolupem el gràfic de la Figura (5), al qual es pot observar una alta correlació entre variables pertanyents al mateix aspecte del joc. És a dir, té sentit que correlin els rebots totals amb els ofensius, o defensius. O que correlin els llençaments intentats i anotats corresponents a una mateixa puntuació. Tot i correlar, es tracta de variables que aporten informació única i rellevant al nostre problema, per tant les mantindrem igualment.

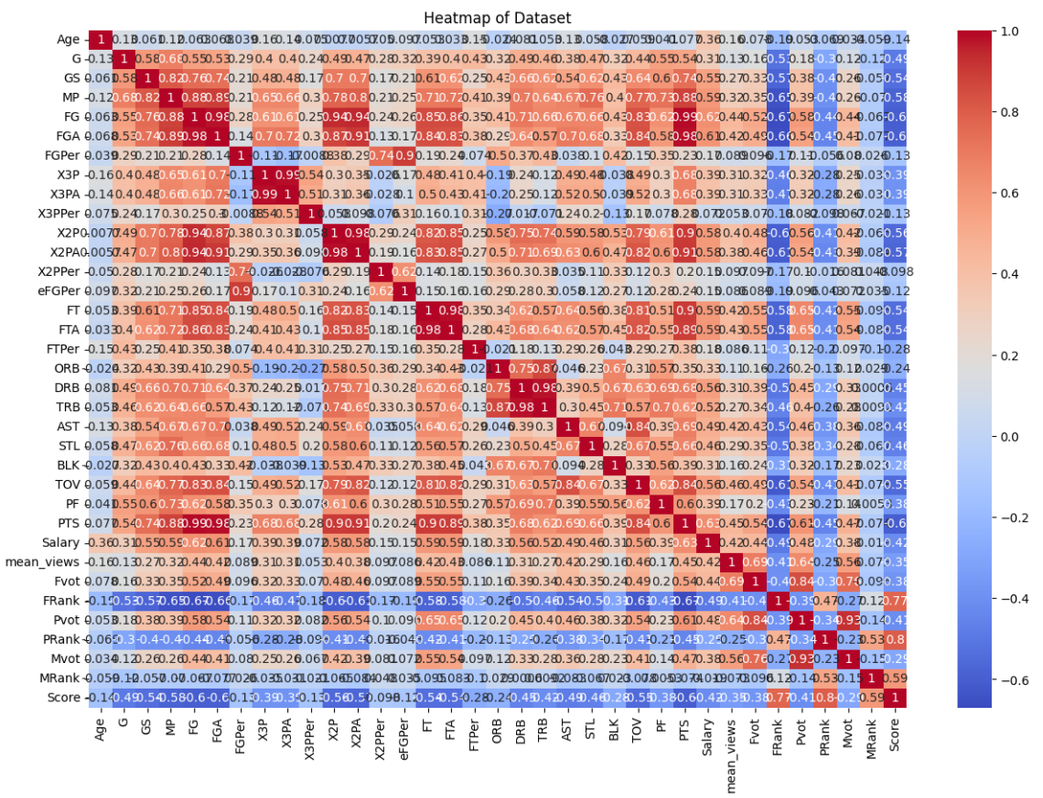

Figura 5: Heatmap de la matriu de correlació

En quant a la distribució de les dades un cop normalitzades, la representació gràfica de les diferents variables es pot trobar a la Figura (6).

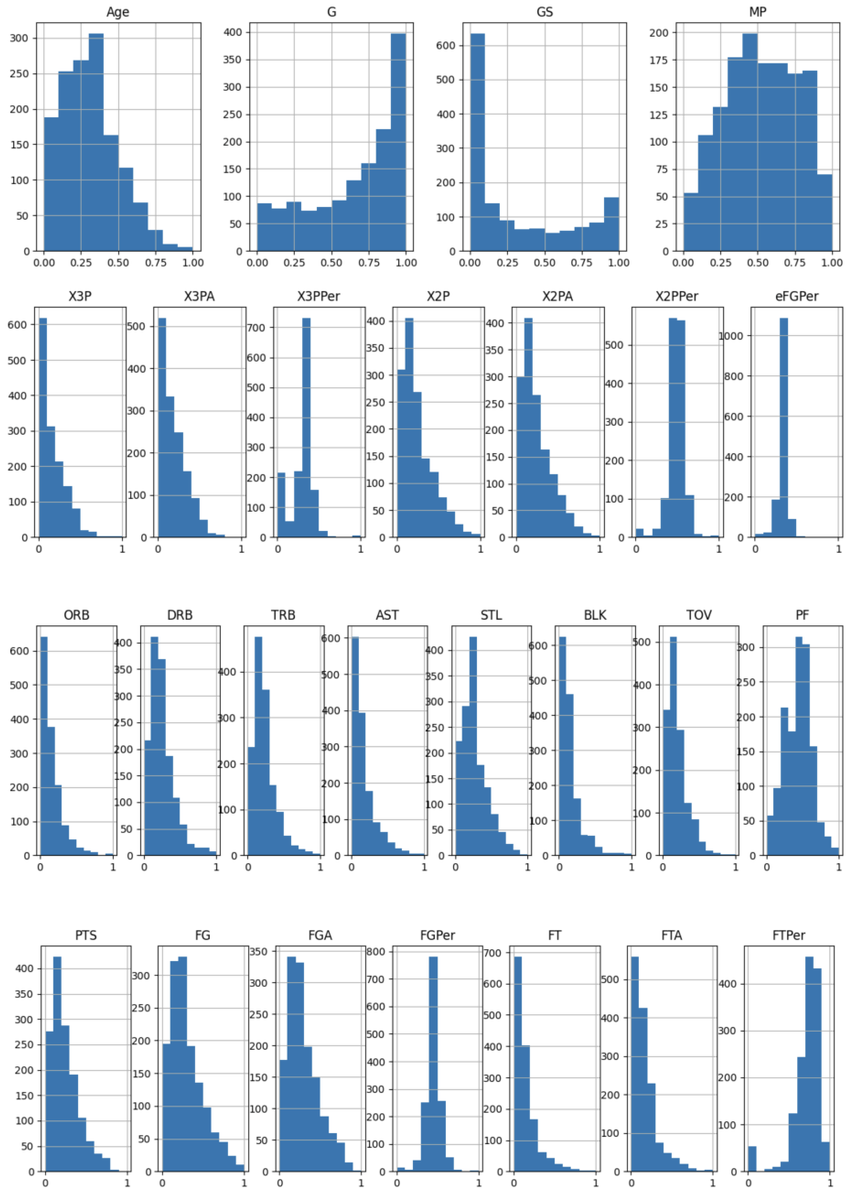

Figura 6: Histogrames de les variables normalitzades

És fàcil intuir com es distribuiria la nostra variable resposta si la representem en relació a les diferents estadístiques, doncs els jugadors que en tenen valors més alts a cadascuna acostumen a ser seleccionats per l'All Star. En canvi, tenim diferents variables sobre les quals voldriem veure com es distribueix el target i la seva influència. Aquestes són el salari del jugador, les seves *mean views*, l'equip on juga i la seva edat. Utilitzant una còpia del nostre dataset en la qual les variables mencionades que són continues estan categoritzades, podem dur a terme la representació.

A partir de les figures adjuntes a continuació podem extreure vàries deduccions. En primer lloc, si ens fixem a la Figura (7), veiem que els jugadors amb salari més alt tenen una major proporció de seleccionats que no pas els que cobren menys diners. A la Figura (8) observem que passa el mateix amb les visites mitjanes. Té sentit, doncs els jugadors més bons i més virals a xarxes acostumen a cobrar un sou més alt, i són el tipus de jugador que acostuma a sortir escollit per aquest partit.
Considerant ara les Figures (9) i (10), podem comprobar com, tal com s'esperava, els jugadors en edats mitjanes i d'equips millors classificats en aquella temporada tenen major proporció de seleccionats. Aquest últim fet s'explica a partir de que la posició en que classifica un equip es valora considerablement en la votació. D'altra banda, és considerat per la major part dels aficionats que un jugador arriba al seu pic de nivell en aquestes edats mitjanes. Per tant, també té sentit aquesta distribució.

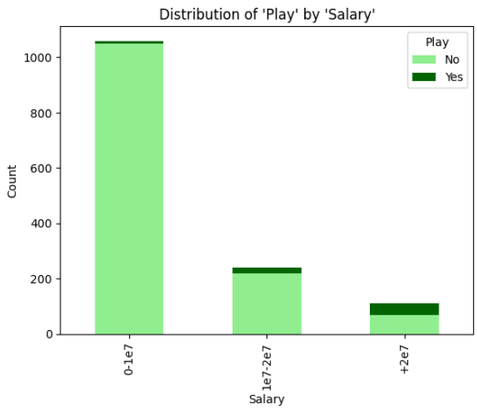

Figura 7: Distribució del target segons *Salary*

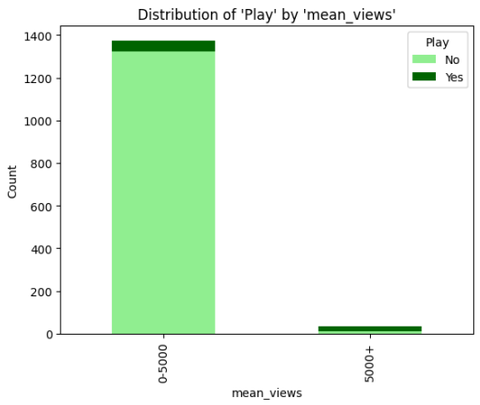

Figura 8: Distribució del target segons *mean_views*

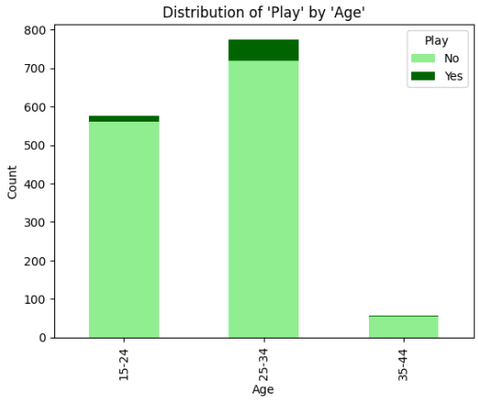

Figura 9: Distribució del target segons *Age*

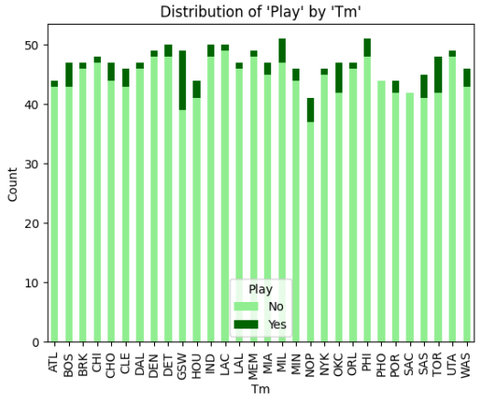

Figura 10: Distribució del target segons *Tm*

3 MODELATGE

Un cop tractades les dades, en aquesta part treballarem amb el dataset corresponent a la temporada 2016-2017, la qual consta de 436 instàncies i 78 atributs. El nombre d'atributs s'ha vist considerablement incrementat degut a la codificació d'atributs realitzada prèviament.

Per poder trobar mesures objectives de la eficiència de cada algorisme que implementem, realitzem la partició de les nostres dades en un conjunt de training (75%) i un conjunt de test(25%), mantenint un balanç entre les dades del target a cada conjunt, evitant així que per aleatorietat el desequilibri fos encara major. És important destacar que cap de les dades al conjunt de test poden ser utilitzades per a escollir el nostre model final, ja que fer això ens donaria un avantatge que no tindrà el nostre model si s'aplica al món real. Per fer el càlcul de les mètriques d'evaluació, realitzem una segona partició, aquest cop al training test, en un conjunt de training (75%) i un de validació (25%). Les mètriques d'evaluació que utilitzarem pels diferents mètodes a implementar seran les següents:

- $Accuracy = \frac{\sum_c tp_c}{n}$, on tp_c és el nombre de positius de cada classe. Aquesta métrica és la proporció de valors que classifiquem correctament. És sensible a classes molt diferents en nombre, com en el nostre cas.
- $Precision_c = \frac{tp}{tp+fp}$, sent tp els positius reals i fp els falsos positius de la classe en concret. Aquesta métrica mesura com de correctament prediu el model a la classe en concret.

- $Recall_c = \frac{tp}{tp+fn}$, sent fn els falsos negatius de la classe en concret. Aquesta métrica ens diu com de correctament es mesura a una classe en respecte als valors reals que té la mateixa. És interessant per no tenir prediccions negatives.

- $F1-score_c = \frac{2*precision_c*recall_c}{precision_c+recall_c}$, és la mitjana harmónica de les dues métriques anteriors, serveix per a trobar un balanç entre precisió i recall.

El nostre és un problema de classificació i, per tant, utilitzarem els següents mètodes d'aprenentatge supervisat: Anàlisi discriminant (LDA/QDA), Regressió logística, classificador Naive de Bayes, k-NearestNeighbours, Arbres de decisió i Random Forest [VMR23]. Per a això, ens ajudarem de diferents paquets que ens ofereix Python [Sci].

3.1 Anàlisi discriminant

Mitjançant l'anàlisi del discriminant podrem entrenar un model de classificació de dades en una sèrie de grups. En aquest cas, voldrem classificar les observacions en els dos grup de selecció. Aquesta metodologia assumeix que la distribució de les dades condicionades a la classe a la que pertanyen segueix una distribuci ó normal multivariant. Segons la relació entre les matrius de covariància per a cada classe, distingirem entre dos tipus d'anàlisi del discriminant: lineal (si podem considerar que les matrius de covariància són iguals per totes les classes) o quadràtica (si no podem fer aquesta assumpció).

En el nostre cas, la variança entre els jugadors no seleccionats per l'All Star és major que la variança entre els que sí ho han estat. Per tant, assumim diferència entre les matrius de covariància, de manera que esperem millors resultats si utilitzem QDA.

En l'ajust del nostre model QDA utilitzem un paràmetre regulador. La funció principal d'aquest és controlar l'equilibri entre la precisió i l'estabilitat del model, a més de reduir l'overfitting. Fent això, garantitzem una major generalització del model. Per tal de trobar el millor valor per aquest paràmetre, ajustem un model per cada possibilitat d'entre les següents: {0, 0.0001, 0.001, 0.01, 0.1, 0.5, 1}. Acte seguit, calculem les mètriques d'evaluació per cadascun i trobem que el paràmetre no ens afecta. De manera arbitrària escollim el valor de 1. Per consultar la taula de mètriques completa, consultar els annexos.

Un cop escollit el valor, ajustem el model quadràtic i obtenim les següents mesures de qualitat, i matriu de confusió sobre el nostre conjunt de validació:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.939024	0.484277	0.469512	0.5

		Predicció	
		0	1
Target	0	77	0
	1	5	0

Com es pot veure, aquest model no prediu cap jugador com a All Star, cosa que és indicativa de que no ens proporciona bons resultats. Provant d'utilitzar validació creuada amb k=5, les mètriques obtenides presenten millors resultats:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.948019	0.526614	0.573939	0.525

3.2 Regressió logística

La regressió logística la utilitzarem per predir la probabilitat de pertinença a una classe binària. Es basa en un model logístic que fa servir una funció d'enllaç per transformar la combinació lineal de variables predictores en una probabilitat. En el nostre cas, la funció de logit link. A partir d'aquesta probabilitat, es pot establir un llindar per fer la classificació final.

En primer lloc, crearem un estimador de regressió logística utilitzant validació creuada amb k=5, on l'objectiu és buscar el millor valor de regularització utilitzant una quadrícula de 20 valors diferents de C, fent servir l'*accuracy* com a mètrica d'avaluació durant la validació creuada. Amb l'estimador, podem trobar quin és el valor òptim de C pel nostre model. En el nostre cas trobem que el millor valor és $C = 0.0001$. Es tracta d'un valor molt petit i que, per tant, correspon a una regularització més robusta.

Un cop escollit el valor de C, ajustem un model de regressió logística i entrenem les nostres dades seguint un procés de validació creuada amb k=5, obtenint les mesures de qualitat mostrades a sota, juntament amb la matriu de confusió sobre el conjunt de validació per aquest mètode. Es pot veure com no prediu cap jugador com a All Star. Com hem mencionat a l'apartat anterior, això és un indicador de que no es proporcionarà bones prediccions.

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.946939	0.48636	0.473469	0.5

		Predicció	
		0	1
Target	0	77	0
	1	5	0

Els resultats del Quadratic Discriminant Analysis i la Logistic Regression són un exemple clar de per què no ens fixem en la "*accuracy*" del nostre algorisme. Fixant-nos en la seva matriu de confusió, veiem que els algorismes classifiquen a totes les mostres com a no participants. Per tant, com que la gran majoria de mostres no participen al partit, el métode ens dona una "*Accuracy*" de més del 90%. Tot i això, les seves métriques "*Precision*" i "*Recall*" són molt baixes per aquests motius. A més, el seu càlcul ens genera errors degut a la divisió entre 0 que es produeix quan es calcula la "*Precision*" de la classe de participants. Per això, podem descartar aquests dos métodes com a algoritmes efectius. És important destacar que aplicant validació creuada sobre QDA sembla que alguns dels participants es classifiquen correctament, peró tot i així, els valors de les métriques d'avaluació són molt baixos.

3.3 Classificador Naive de Bayes

El classificador Naive Bayes és un algorisme basat en el teorema de Bayes que considera que les característiques són independents entre si tenint en compte el valor de la classe. Aquesta simplificació permet una classificació ràpida i eficient, especialment en conjunts de dades amb moltes característiques com és el nostre.

A l'ajustar un model seguint aquest mètode, i entrenar les dades seguint un procés de validació creuada amb k=5, obtenim els valors a les mètriques mostrats a sota, juntament amb la matriu de confusió sobre el conjunt de validació. Podem observar com no prediu gaire bé els jugadors no seleccionats, doncs 12 que haurien d'estar classificats com a tal els agrupa contràriament.

Accuracy	F1 Macro	Precision Macro	Recall Macro
0.857143	0.525784	0.522377	0.562874

		Predicció	
		0	1
Target	0	65	12
	1	3	2

Observant la matriu de confusió, veiem que ara si que es classifiquen millor els resultats, i tenim alguns jugadors del partit de les estrelles que són classificats com a tal. Per tant, sembla que tot i que aquest métode té una "*accuracy*" més baixa, pot ser més desitjable ja que a vegades classifica correctament a participants. Tot i això, veiem que realment el seu "$F-Score$" és més baix que l'obtingut per QDA i validació creuada, per tant, aquest métode tampoc és gaire bo.

3.4 kNN

k-Nearest Neighbors (kNN) és una tècnica que es basa en la idea que els punts de dades similars tendeixen a estar a prop a l'espai de característiques. A kNN, la classificació d'un punt de dades es determina per la majoria de vots dels seus k veïns més propers.

En primer lloc, realitzarem una cerca exhaustiva dels millors hiperparàmetres pel nostre model. En aquest cas, provarem diferents valors tant pel nombre de veïns com per les mètriques de distància. Pel nombre de veïns escollirem entre els següents: {1, 3, 5, 7, 10, 15, 20}. Mentre que per la mètrica de distància entre les següents: {*Euclidean, Minkowski, Manhattan*}.
Tenint en compte un global de totes les mesures de qualitat obtingudes per a cada combinació, determinem que el millor nombre de veïns que podem escollir és k=1. Dins d'això, veiem que la mètrica de distància que s'ajusta millor, per sobre de les altres dues, és la distància de *Manhattan*. Per veure els resultats complerts, es pot trobar una taula amb aquests als annexos.

Un cop escollits els hiperparàmetres, ajustem el model i entrenem les dades seguint un procés de validació creuada amb k=5. Podem observar com és, de moment, el model que millor s'ajusta a les nostres dades i ens proporciona els millors resultats. Això es pot veure tant a les mètriques obtingudes com a la matriu de confusió sobre el conjunt de validació:

Accuracy	F1 Macro	Precision Macro	Recall Macro
0.972354	0.785669	0.854113	0.756694

		Predicció	
		0	1
Target	0	77	0
	1	4	1

Per al cas de KNN, mitjançant validació creuada i la cerca d'hiperparàmetres, hem obtingut una matriu de confusió acceptable, ja que classifica correctament a tots els jugadors que no han participat al partit de les estrelles. Tot i això, veiem que no es classifica tan bé els jugadors que no hi participen. Tot i això, el valor que obtenim fent servir validació creuada ens aporta unes métriques força altes. Veiem que la "*precision*" és molt alta, probablement degut a que la gran majoria de mostres predites com a jugadors del partit són jugadors que realment han estat seleccionats. També podem assenyalar que el "*recall*" és prou alt, indicador de que la majoria de jugadors de cada classe es separen a la que els hi toca. Justament per aquest motiu, tenim un valor tant alt de "$F1Macro$". Això indica que realment el nostre model classifica bé les dades.

3.5 Decision Tree i Random Forest

Amb l'arbre de decisió construirem un model en forma d'estructura d'arbre per prendre decisions on cada node intern representa una característica, cada branca representa una regla de decisió i cada full representa el resultat de la classificació. Amb Random Forest buscarem millorar la precisió i reduir el sobreajustament, prenent decisions basades en la classificació de múltiples arbres.

3.5.1 Decision Tree

En primer lloc, ajustem un *DecisionTreeClassifier* amb les nostres dades de training. L'arbre de decisió entrenat el podem representar gràficament de la següent manera:

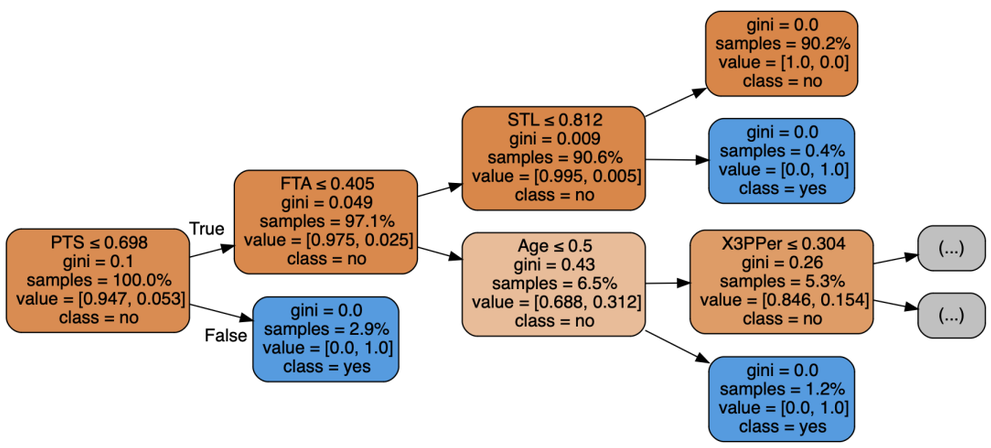

Figura 11: Arbre de decisió entrenat per defecte sobre el training set

A l'ajustar amb l'arbre per defecte, és interessant veure les decisions que li afecten. Veiem com sorprenentment, el nombre de tirs lliures intentats és el primer criteri de selecció. De fet, tots els jugadors que tiren bastants tirs lliures són tots jugadors del All-Star. El següent criteri és més intuitiu, i és el nombre de visites de la seva pàgina de la Wikipedia. Podríem seguir apronfundint en l'arbre, però podem assumir que la resta de criteris no són tant importants, ja que si ho fossin, serien escollits abans per l'algorisme.

Si calculem les dimensions d'aquest, ens dona que l'arbre té 13 nodes, i una profunditat de 5 nivells. Aquesta última mesura ens diu que l'arbre de decisió és poc profund i no hauria de donar gaires bons resultats. Predint amb aquest model d'arbre de decisió per defecte, sobre el conjunt de validació, obtenim les següents mètriques i matriu de confusió:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.914634	0.658942	0.64693	0.674026

		Predicció	
		0	1
Target	0	73	4
	1	3	2

Veiem com els resultats de l'arbre no són gaire bons, ja que es prediuen molts jugadors com a participants de l'All Star quan no ho són, i es repeteixen errors d'altres models. Tot i això, podem considerar aquest valor com a millor que Logistic Regression o QDA, ja que és capaç de classificar correctament a valors com a participants prou eficientment. Aquest métode té una métrica de "*F1 Macro*" moderadament alta. Com el mètode emprat no ens proporciona molt bons resultats, tal com hem fet anteriorment en altres seccions, realitzarem una cerca exhaustiva d'hiperparàmetres per aquest tipus

de model utilitzant validació creuada amb k=5. Provarem els següents hiperparàmetres amb les seves respectives opcions:

- Criteri utilitzat per evaluar la qualitat d'una partició: {*gini*, *entropy*}.
- Nombre mínim de mostres per dividir un node intern: {1,2,3,4,5}
- Nombre mínim de mostres requerides a un node: {1,2,3,4,5}
- Nombre màxim de features a considerar en cada divisió: {*auto, sqrt, log2, None*}

Utilitzant com a mètrica d'evaluació principal la mitja l'F1-score per les classes per separat, procedim amb la cerca, trobant que els millors hiperparàmetres pel nostre model basat en un arbre de decisió són:

{criteri: *gini*, max_depth: *None*, max_features: *auto*, min_samples_leaf: 1, min_samples_split: 4}.

Per veure la taula amb els resultats per a cada combinació, consultar els annexos.
Entrenant ara un model amb aquests valors, obtenim les següents mètriques i matriu de confusió:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.914634	0.658942	0.64693	0.674026

		Predicció	
		0	1
Target	0	75	2
	1	2	3

Fent la validació creuada per a trobar els millors hiperparàmetres, acabem obtenint exactament els mateixos resultats. Això ens pot indicar que utilitzar un sol arbre de decisió pot ser insuficient, per molt bo que sigui.

3.5.2 Random Forests

Tal com hem fet anteriorment, entrenem un primer classificador de Random Forests amb les dades de training. Amb aquest, obtenim les següents mètriques i matriu de confusió:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.939024	0.706093	0.730769	0.687013

		Predicció	
		0	1
Target	0	75	2
	1	3	2

A continuació, ajustarem un altre *RandomForestClassifier*, però assignant pesos a les classes de manera inversament proporcional a la seva freqüència a les dades d'entrenament. És a dir, introduint un balanç. Seguint el mateix procediment, en aquest cas tenim les següents mesures de qualitat i matriu de confusió:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.926829	0.605769	0.64135	0.587013

		Predicció	
		0	1
Target	0	75	2
	1	4	1

En busca de millor rendiment, a l'igual que amb anteriors mètodes, realitzarem una cerca exhaustiva de valors pels diferents hiperparàmetres, que en aquest cas seran: nombre d'arbres = {200, *None*}, màxima profunditat = {100, *None*}, nombre mínim de mostres per dividir un node = {4,6}, nombre mínim de mostres requerides a un node = {4,6} i balanç = {*None, balanced, balanced_subsample*}. En aquesta cerca, obtenim que la millor combinació d'hiperparàmetres és la següent:

{balanç: *balanced*, max_depth: 100, min_samples_leaf: 4, min_samples_split: 4, n_arbres: 200}

Es pot trobar una taula amb els resultats de la cerca per diferents combinacions als annexos.
Amb aquests hiperparàmetres ajustem un model i obtenim els següents resultats:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.939024	0.756387	0.736842	0.780519

		Predicció	
		0	1
Target	0	74	3
	1	2	3

En aquest cas veiem que realment la unió de diferents arbres ens aporta bons resultats, ja que ara la "*F1 Macro*" és prou alta per a ser l'algorisme per defecte. De fet, utilitzant el mètode amb "balance", obtenim uns resultats pitjors.

Tornant a executar seguint un procés de validació creuada per a escollir es millors hiperparàmetres possibles, tenim una puntuació de "$F1Macro$" bastant alta, tot i que segueix sent lleugerament inferior al valor per KNN. De fet, només supera a aquest mètode per "*Recall*".

3.5.3 Extra trees

Extra trees és una variant del Random Forest que es diferencia en que realitza divisions aleatòries a cada node de l'arbre, en lloc de buscar les millors divisions, cosa que ho fa més ràpid però menys precís. A més, utilitza una major quantitat d'arbres i fa mitjanes ponderades per prendre les decisions finals.

Tal com hem fet amb els dos mètodes anteriors, entrenarem un primer model per defecte i després realitzarem una cerca exhaustiva d'hiperparàmetres amb l'objectiu de maximitzar les mètriques i millorar el rendiment. El primer model ajustat d'Extra Trees ens genera els següents resultats:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.95122	0.737179	0.814346	0.693506

		Predicció	
		0	1
Target	0	76	1
	1	3	2

En aquest cas, al ser una variant del Random Forest, els possibles hiperparàmetres són els mateixos. Realitzarem una petita modificació de totes maneres, establint que el nombre d'arbres sigui o *None*, o 150 (en comptes de 200). La millor combinació que podem obtenir un cop realitzada la cerca és la següent:

{balanç: *balanced*, max_depth: *None*, min_samples_leaf: 2, min_samples_split: 4, n_arbres: 150}

Tal com amb els altres mètodes, la taula amb les mètriques per les diferents combinacions pot trobar-se als annexos.

Ajustant un *ExtraTreesClassifier* amb aquests hiperparàmetres obtenim els següents resultats:

Accuracy	F1 Macro	Precision Macro	Recall Macro
0.939024	0.756387	0.736842	0.780519

		Predicció	
		0	1
Target	0	74	3
	1	2	3

Utilitzant ara Extra Trees, veiem que el model per defecte ens aporta els tercers millors resultats fins al moment, cosa que indica que aquest métode pot ser molt eficient per al model. Tornant a repetir el procés, trobem els millors hiperparàmteres per al nostre model, i observem que tornem a obtenir unes bones métriques. Tot i això, aquestes continuen sent lleugerament inferiors a les del KNN.

4 ANÀLISI DELS RESULTATS I SELECCIÓ DEL MODEL

	Accuracy	F1 Macro	Precision Macro	Recall Macro
KNN	0.972354	0.785669	0.854113	0.756694
RF-best	0.939024	0.756387	0.736842	0.780519
extra_trees-best	0.939024	0.756387	0.736842	0.780519
extra_trees	0.95122	0.737179	0.814346	0.693506
RF-default	0.939024	0.706093	0.730769	0.687013
DT-default	0.914634	0.658942	0.64693	0.674026
DT-best	0.914634	0.658942	0.64693	0.674026
RF-balance	0.926829	0.605769	0.64135	0.587013
QDA	0.948019	0.526614	0.573939	0.525
GaussianNB	0.857143	0.525784	0.522377	0.562874
Logistic Regression	0.946939	0.48636	0.473469	0.5

Figura 12: Taula de mètriques dels diferents models ajustats

En aquesta secció, observarem el rendiment cadascun dels models que hem fet servir, i triarem el que creiem que ens aporta els millors resultats a l'aplicar la validació.

Com ja hem indicat anteriorment, per al nostre cas, no busquem la millor "*accuracy*" en la predicció, ja que la diferéncia de mostres entre les dues classes és molt gran, i per tant, aquesta métrica no ens aporta una bona referéncia de la eficiéncia de cada model. En canvi, farem servir la "$F1 - Score$", ja que aquest indicador és la mitjana harmónica entre "*precision*" i "*recall*", i per tant, troba un balanç entre aquestes dues. Ens interessa la métrica "*precision*" ja que aquesta ens indica el ràtio entre positius correctes que prediu el nostre model i el nombre total de positius que prediu. Per tant, una baixa precisió indicaria que estem predint a molts jugadors que no van jugar com a participants del All-Star o a molts jugadors del partit com a no participants.
La altra métrica interessant és el "*recall*", aquesta ens indica com de correctament es mesura una classe, és a dir, el nombre de jugadors ben classificats. Per tant, en el nostre cas, un baix recall ens indicaria que molts participants del All-Star están sent classificats com a no participants o viceversa. És interessant veure la diferéncia entre aquests dos valors, ja que el "*recall*" es fixa més en si la classe està sent ben classificada i la "*precision*" es fixa en si les prediccions están ben classificades.
Com que en el nostre cas ens interessen les dues classes per igual, no ens fixarem en la "$F1 - score$" d'una de les dues classes, si no que farem servir la seva mitjana entre les dues classes, la "$F1 - score\ Macro$".

A la taula adjunta al principi de la secció es poden trobar tots els mètodes emprats, amb les seves respectives mesures de qualitat, ordenats per major *F1 Macro*.

Havent observat a tots els models i les métriques que ens aporten al fer la seva validació, veiem com el millor algorisme que podem escollir amb la informació que tenim ara mateix és el de KNN, ja que aquest és el que té la millor "$F-Score$", que ja hem comentat que és la métrica que volem maximitzar i més ens interessa.

5 CONCLUSIONS

A l'escollir el nostre model, realitzem 3 prediccions diferents. Una sobre la part de la nostra temporada que hem separat al principi del modelatge, és a dir, la part de "test". Per altra banda, predim sobre les dues següents temporades, per a comprovar si els criteris del nostre model canvien amb el temps.

5.1 Testing

Fent la predicció sobre aquelles dades que hem apartat a l'inici del nostre procés, obtindríem les següents mètriques i matriu de confusió:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.963303	0.823625	0.823625	0.823625

		Predicció	
		0	1
Target	0	101	2
	1	2	4

Observem com el nostre model prediu correctament la gran majoria de valors dels no participants, cosa que és molt positiva. A més, prediu correctament 4 dels 6 seleccionats per l'All Star. Aquests resultats són encara millors dels que hem obtingut sobre les mostres de validació. Aquest és un fet positiu, ja que ens indica que el nostre model no esta caient en $overfitting$, i pot estimar bastant eficientment mostres que desconeix completament.

5.2 Estabilitat del model

Per a veure si la selecció dels jugadors és un criteri més o menys estable, és interessant veure si podem predir els seleccionats dels següents anys. Si predim aquestes mostres amb el model escollit, obtenim els següents resultats:

Per la temporada 17/18:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.959839	0.762223	0.774184	0.751396

		Predicció	
		0	1
Target	0	466	9
	1	11	12

Per la temporada 18/19:

Accuracy	**F1 Macro**	**Precision Macro**	**Recall Macro**
0.959916	0.787329	0.820088	0.761418

		Predicció	
		0	1
Target	0	441	7
	1	12	14

Observem que el nostre model pot predir amb una eficiència similar altres temporades. Això ens indica que el criteri del All-Star no ha variat, o que almenys és similar entre temporades consecutives. A més, veure que la F1 Macro incrementa a la tercera temporada (la 18/19) pot mostrar que la relació no disminueix com més allunyada estigui la temporada de la que hem fet servir per a estimar el model.

5.3 Variables més importants

Un aspecte interessant a destacar és la gràfica que indica la importància de les diferents variables sobre el model de Random Forests amb hiperparàmetres escollits per validació creuada, que cal recordar que és un dels millors models que hem trobat. La gràfica és la següent:

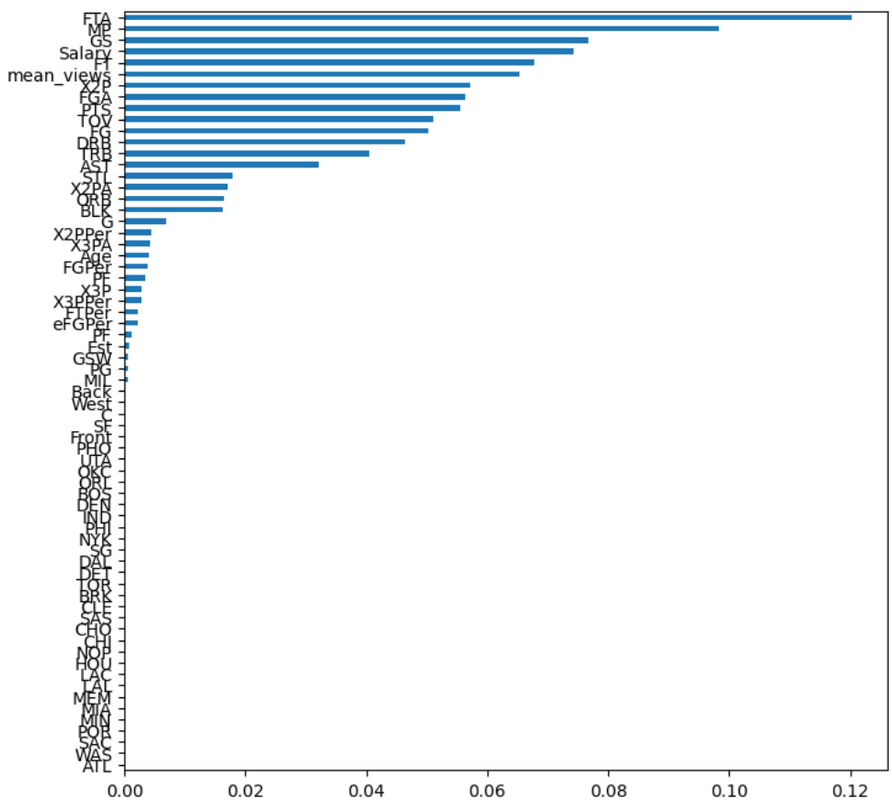

Figura 13: Importància de cada variable al model RF-best

Sorprenentment, veiem que la variable que més influeix sobre el model és la de tirs lliures intentats. Això és bastant sorprenent, sobretot tenint en compte que està per sobre de variables com punts anotats, salari o asisténcies. La causa d'això pot residir en que els jugadors que més tirs lliures llencen, són els que tenen tendència a tenir més la pilota i prendre tirs més forçats que acaben en falta personal. És a dir, les estrelles dels equips.

També és interessant veure que el nombre de partits com a titular és molt més important que el nombre de partits jugats, això és perque hi han molts jugadors que juguen molts partits, peró no són tants els que surten com a titulars a tants partits. Podem observar també que realment el nombre de visites a la Wikipedia és realment important, indicador de que la popularitat del jugador importa més que els punts anota.

Un fet interessant és veure que jugar a la conferència est aporta una probabilitat major de jugar el partit que fer-ho a la oest. És un fet dubtós, ja que el partit és disputat pel mateix nombre de jugadors de cada conferència. Una explicació possible és que les estadístiques dels jugadors de l'oest siguin superiors a les de l'est, però com que han de seleccionar el mateix nombre per a cada conferència, aquests tinguin una probabilitat major de ser seleccionats siguin com siguin les seves estadístiques.

5.4 Limitacions i possibles millores

Creiem que el nostre model és prou encertat, sobretot tenint en compte que la elecció dels jugadors es fa a partir de votacions, i aquestes tenen òbviament una component subjectiva. Per tant, molts dels criteris que pot seguir la gent al votar no seran constants. A més, hi ha una component no mesurable numèricament que influeix en la opinió de qui vota, com l'estil del jugador o com es comporta a la pista.

Tot i això, creiem que podríem fer millores al nostre model si tinguéssim accés a més dades, i poder trobar resultats encara millors. Per exemple, podríem anar entrenant el nostre model amb noves dades de noves temporades. D'aquesta manera, obtindríem més mostres, ja que a l'haver-hi tan sols poc més de 20 seleccionats per temporada, el model no té prous referéncies per a augmentar la seva precisió.

Un altra possibilitat seria tenir en compte la posició on van quedar els equips a aquella temporada, cosa que indica si l'equip ha tingut éxit i si, per tant, els jugadors que hi juguen són més valorats, ja que hem vist anteriorment que aquest fet afecta al resultat final. Una altra variable interessant és el nombre d'habitants de la ciutat o estat d'on és l'equip, ja que això ens pot indicar com de famosos són els equips. Altres mesures de popularitat serien els seguidors a les xarxes socials de l'equip i el jugador. Aquestes últimes variables defineixen el tipus de mercat al que pertany l'equip que se'n diu. Es considera que equips amb un mercat més gran, com podrien ser els Lakers, més fàcilment tindran jugadors seleccionats si l'equip està mitjanament ben posicionat en la classificació. En canvi, equips de mercats més petits, com els Kings, han d'estar en llocs molt alts a la classificació per a que a algun dels seus jugadors se'l tingui en consideració a la votació.

Una possible limitació és que el partit es disputa a mitja temporada i, per tant, no tenim les dades de tota la temporada com hem tingut al modelar. Conseqüentment, les prediccions potser no serien tant precises, tot i que seguirien sent possibles de fer, ja que es fan tenint en compte la mitjana de cada estadística. Realment no sabem si aquest fet seria millor o pitjor, donat que la gent que votés només hauria vist les estadístiques fins a aquest punt.

Les limitacions ja les hem comentat abans, el procés de selecció és subjectiu i, per tant, té una part aleatória molt difícil de modelar. Donat això, potser farien falta dades de temporades anteriors per a poder fer-nos una idea de la opinió general que es té d'un jugador.

6 BIBLIOGRAFIA

Referències

[VMR23] Marta Arias Vicente, Luis Antonio Belanche Muñoz, and Alexis Molina Martinez De los Reyes. *Notebooks i transparències del curs d'Aprenentatge Automàtic 1*. Raco-FIB, 2023.

[Bas] BasketballReference. *Glossary—BasketballReference*. URL: `https://www.basketball-reference.com/about/glossary.html`.

[Sci] Scikit-Learn. *Scikit Learn - Machine Learning in Python*. URL: `https://scikit-learn.org/stable/index.html`.

7 ANNEXOS

Taules de combinacions pels hiperparàmetres dels models

model	reg	**Recall class 1**	**F1 class 1**	Accuracy	F1 Macro	Precision Macro	Recall Macro
QDA	0.0000	0.0	0.0	0.939024	0.484277	0.469512	0.5
	0.0001	0.0	0.0	0.939024	0.484277	0.469512	0.5
	0.0010	0.0	0.0	0.939024	0.484277	0.469512	0.5
	0.0100	0.0	0.0	0.939024	0.484277	0.469512	0.5
	0.1000	0.0	0.0	0.939024	0.484277	0.469512	0.5
	0.5000	0.0	0.0	0.939024	0.484277	0.469512	0.5
	1.0000	0.0	0.0	0.939024	0.484277	0.469512	0.5

Mètriques segons el paràmetre regulador del QDA

param_n_neighbors	param_metric	mean_test_accuracy	mean_test_f1_macro	mean_test_precision_macro	mean_test_recall_macro
1	manhattan	0.972354	0.785669	0.854113	0.756694
1	euclidean	0.969324	0.773744	0.834113	0.755081
1	minkowski	0.969324	0.773744	0.834113	0.755081
3	euclidean	0.963263	0.709364	0.862957	0.681720
3	manhattan	0.963263	0.709364	0.862957	0.681720
3	minkowski	0.963263	0.709364	0.862957	0.681720
5	euclidean	0.963263	0.706203	0.881369	0.658333
5	manhattan	0.963263	0.706203	0.881369	0.658333
5	minkowski	0.963263	0.706203	0.881369	0.658333
7	manhattan	0.954079	0.594818	0.676875	0.575000
7	minkowski	0.948019	0.526614	0.573939	0.525000
7	euclidean	0.948019	0.526614	0.573939	0.525000
10	minkowski	0.944988	0.485851	0.472494	0.500000
15	minkowski	0.944988	0.485851	0.472494	0.500000
20	minkowski	0.944988	0.485851	0.472494	0.500000
20	euclidean	0.944988	0.485851	0.472494	0.500000
15	euclidean	0.944988	0.485851	0.472494	0.500000
10	euclidean	0.944988	0.485851	0.472494	0.500000
10	manhattan	0.944988	0.485851	0.472494	0.500000
15	manhattan	0.944988	0.485851	0.472494	0.500000
20	manhattan	0.944988	0.485851	0.472494	0.500000

Mètriques pels hiperparàmetres del kNN

param_criterion	param_max_depth	param_max_features	param_min_samples_leaf	param_min_samples_split	mean_test_f1_mac	mean_test_f1_class_0	mean_test_f1_class_1
gini	None	sqrt	1	4	0.758427	0.964631	0.552222
gini	20	sqrt	1	5	0.758427	0.964631	0.552222
gini	10	sqrt	1	4	0.758427	0.964631	0.552222
gini	20	auto	1	4	0.758427	0.964631	0.552222
gini	20	auto	1	5	0.758427	0.964631	0.552222

Mètriques pels hiperparàmetres del Decision Tree

param_max_depth	param_min_samples_leaf	param_min_samples_split	mean_test_f1_mac	mean_test_f1_class_0	mean_test_f1_class_1	mean_test_acc
100	4	4	0.819184	0.985035	0.653333	0.971429
None	4	4	0.819184	0.985035	0.653333	0.971429
100	4	4	0.819184	0.985035	0.653333	0.971429
100	4	6	0.819184	0.985035	0.653333	0.971429
None	4	4	0.804752	0.982838	0.626667	0.967347

Mètriques pels hiperparàmetres del Random Forest

param_max_depth	param_min_samples_leaf	param_min_samples_split	mean_test_f1_mac	mean_test_f1_class_0	mean_test_f1_class_1	mean_test_acc
None	2	4	0.805242	0.977151	0.633333	0.957203
None	2	4	0.805242	0.977151	0.633333	0.957203
100	2	4	0.805242	0.977151	0.633333	0.957203
100	2	4	0.805242	0.977151	0.633333	0.957203
None	2	6	0.789168	0.984051	0.594286	0.969371

Mètriques pels hiperparàmetres de l'Extra Trees