Sfruttamento dei Dati di Prestazione per la Classificazione delle Posizioni, la Regressione dei Valori di mercato e il Clustering dei Calciatori

Tommaso Menghini matricola 864946 Tommaso Pozzi matricola 864654

15 Luglio 2024

Abstract

Questo studio si basa sulla convinzione che un approccio data-driven possa essere fondamentale nell'ottenere un vantaggio competitivo nel gioco del calcio. In questo caso si è voluto dimostrare che attraverso la misurazione e la quantificazione di alcuni aspetti del gioco per ogni calciatore si possa determinarne il valore di mercato, classificarne il ruolo e trovare cluster in grado di raggruppare giocatori simili. Per il task di regressione si è optato per algoritmi come il Support Vector Regressor, il KNN e i Processi Gaussiani al fine di stimare valori di mercato. Per la classificazione della posizione di un giocatore si è pensato al Support Vector Machine e al Random Forest, arrivando a quasi il 90% di accuracy. Per il task di clustering si è fatto uso dell'algoritmo delle k-medie e dei modelli di mistura gaussiani, ottenendo risultati differenti dalle aspettative.

1 Introduzione

FBref.com[2] è un sito web dedicato al tracciamento delle statistiche relative ai calciatori e alle squadre di calcio di tutto il mondo. Il focus dell'analisi è stato posto sui protagonisti del calcio, cioè i suoi interpreti principali: i giocatori. Importante è sottolineare che come riferimento si è preso l'ultima stagione calcistica 2023-2024, da poco conclusa, dei top 5 campionati europei: "Bundesliga", "La Liga", "Ligue 1", "Premier League", "Serie A"; cioè i campionati rispettivamente di Germania, Spagna, Francia, Inghilterra e Italia.

Si è quindi considerato un'insieme di statistiche per giocatore ricavate tramite *Web Scraping* dal sito sopra citato. FBref [2] è organizzato in tabelle, ad ognuna corrisponde un macro-concetto del gioco come ad esempio il passaggio, il tiro o le azioni difensive. Ad esse, quindi, è associato uno specifico set di statistiche misurate per ogni giocatore. Si sono prese in esame la tabella *"Shooting"*, che riassume statistiche del tiro, *"Defense Action"* che si concentra su aspetti difensivi e la tabella *"Possession"* molto interessante perchè al suo interno vi sono statistiche che approfondiscono il come e il dove un giocatore tratti il pallone; un esempio può essere il numero di tocchi fatti nella trequarti offensiva del campo.

Alle feature prese in considerazione fino a questo punto se ne è aggiunta un'ulteriore corrispondente al valore di mercato corrente del giocatore. Quest'ultima, però è stata ricavata dal sito web Transfermarkt.com [5], il più noto e affidabile per questioni di mercato nel calcio.

Lo studio si articola in 3 differenti task: **Regressione**, **Classificazione** e **Clustering**. Il primo ha come obiettivo quello di valutare il miglior algoritmo per l'imputazione dei cosiddetti *Missing Values*. Nel dataset infatti è presente una variabile con dati mancanti e l'idea è stata quella di utilizzare differenti algoritmi di regressione per imputarli e completarli. La feature in questione è il valore di mercato, ed è stato interessante al fine di dare una valutazione quantomeno qualitativa dell'imputazione proposta, confrontare i valori ottenuti con quelli riportati da siti specializzati e testate giornalistiche, che forniscono valutazioni professionali e aggiornate dei giocatori.

Nel task di classificazione si sono proposti ed esaminati differenti algoritmi con l'obiettivo di identificare quell'algoritmo che meglio si adattasse al problema trattato, cioè quello di prevedere la posizione di un giocatore (delle 4 disponibili) in base al valore assunto dalle feature, fornendo le previsioni più accurate e affidabili.

L'ultimo task è stato quello di Clustering: l'obiettivo è quello di individuare gruppi tra le istanze tali da essere il più possibile eterogenei fra loro ed omogenei al loro interno. Anche in questo caso si è provato ad implementare più di un algoritmo al fine di scegliere quello che portasse al clustering di miglior qualità. Il senso comune può portare a pensare che i gruppi siano figli dei ruoli assegnati ai giocatori, ma ciò non è automatico.

2 Pre-Processing

Il dataset su cui è incentrato lo studio è il risultato di un lavoro di *Web Scraping* svolto su FBref [2] e TransferMarkt [5], reso possibile dalla libreria R *worldfootballR* [6]. In primo luogo si sono estratti i dati da FBref [2] contenuti nelle tabelle citate nel capitolo 1 che poi si sono unite in un primo dataset. Di questo, poi, si sono considerati solo quei record, cioè i giocatori, per cui il valore della variabile *"Mins_Per_90"*, ossia i minuti giocati nella stagione divisi per 90, fosse maggiore della mediana, in ragione al fatto che un calciatore poco impiegato sia non rilevante dal punto statistico e quindi non sia altro che rumore nei dati. In seguito si sono estratti da TransferMarkt [5] i valori di mercato di tutti i giocatori dei top 5 campionati europei. Durante l'analisi del dataset ricavato, si è riscontrato che i dati erano mal formattati: le colonne erano sfalsate, e a volte i nomi dei giocatori erano presenti nella colonna dei valori di mercato e viceversa. Una possibile ragione potrebbe risiedere nella funzione utilizzata per scaricare i dati dalla rete. Per risolvere questo problema, si è implementato una serie di passaggi per la riorganizzazione dei dati in modo da rendere possibile il join con l'altro dataset composto in precedenza.

Prima di unirli, però, c'è stato bisogno di un' ulteriore intervento verso quei giocatori definiti come doppi, cioè quei giocatori nel dataset dei valori che compaiono più di una volta perchè omonimi. Sono stati tre i casi di omonimia. Il primo riguardava i due *"Raùl Garcia"*, tutti e due militano nella lega spagnola, ma uno per l'Osasuna e l'altro per l'Athletìc Bilbao. Entrambi però non risultavano come calciatori sopra la soglia della mediana per minuti giocati, pertanto sono stati rimossi dal dataset. Il secondo caso riguardava i due *"Danilo"*, uno alle forze del Nottingham Forest in Inghilterra e l'altro capitano della Juventus nell'ultima stagione. Entrambi sono ritenuti calciatori statisticamente rilevanti e si è deciso, quindi, di rinominarli aggiungendo al termine del loro nome l'iniziale della squadra in cui giocano, quindi rispettivamente *"DaniloN"* e *"DaniloJ"*. Infine l'ultimo caso di omonimia è stato quello di *"Vitinha"*, uno di questi è il famoso centrocampista sotto contratto al Paris Saint Germain, mentre l'altro milita in Serie A nelle file del Genoa, ma, almeno per quest'ultima stagione, ha ottenuto poco spazio e si è rivelato essere quindi statisticamente irrilevante.

Dopo aver risolto quest'ultima questione si è potuto unire i due dataset ricavati dai due siti web, in modo tale da considerare solamente le istanze presenti nel primo dei due, cioè considerando solamente i giocatori con un valore di *"Mins_Per_90"* maggiore della mediana pari a 12.40. Concludendo questa prima parte di ricerca, estrazione e composizione dei dati, lo studio parte da un numero di istanze pari a 1418 per 66 feature. Si ha pertanto un numero elevato di variabili che, in virtù di come è stato pensato il dataset e di come è organizzato FBref [2], è possibile suddividere in 4 macro-concetti corrispondenti alle informazioni personali dei calciatori e alle informazioni contenute in quelle tabelle il cui significato è stato approfondito nel capitolo 1.

Certamente il ruolo ricoperto da un giocatore è la caratteristica che ne determina maggiormente le peculiarità e la variabile *"Pos"* indica proprio la posizione ricoperta da un determinato calciatore. Come presentata originalmente da FBref [2] può assumere sette diversi valori presentando quindi una granularità elevata. Si è deciso di modificare questo aspetto e di passare ad una granularità inferiore riducendo i valori possibili per questa feature a quattro, compatibili con il senso comune: *"GLK"*, *"DF"*, *"MF"* e *"FW"*, rispettivamente corrispondenti a "Portiere", "Difensore", "Centrocampista" e "Attaccante". Le ragioni di questa scelta sono da ricercare nelle richieste e nelle domande dello studio,

per cui si è ritenuto che il livello di dettaglio precedentemente proposto non fosse necessario.

Gestire i valori mancanti in un dataset è fondamentale nell'analisi dei dati. Infatti la loro presenza può influenzare significativamente i risultati delle analisi statistiche e dei modelli predittivi, portando a conclusioni errate o distorte. Pertanto, è essenziale adottare strategie appropriate per affrontare questo problema. Esistono diverse tecniche per gestire i valori mancanti, che vanno dall'eliminazione delle osservazioni o delle variabili contenenti *missing values*, all'imputazione. Ogni metodo ha i suoi vantaggi e svantaggi, e la scelta della tecnica più adeguata dipende dal contesto specifico e dall'entità del problema. In questo caso si hanno 949 missing values, pochi rispetto alla dimensione del nostro campione, sparsi nelle seguenti variabili:

- *"SoT_percent_Standard"*;
- *"G_per_Sh_Standard "*;
- *"G_per_SoT_Standard"*;
- *"Dist_Standard"*;
- *"npxG_per_Sh_Expected"*;
- *"Cmp_percent_Long"*;
- *"Tkl_percent_Challenges"*;
- *"Tkld_percent_Take"*;
- *"Succ_percent_Take"*;
- *"prezzo_numerico"*;

Tutte, ad esclusione di *"prezzo_numerico"*, sono rapporti tra altre variabili che possono assumere valore nullo. Chiaramente dividere per 0 porta ai missing values, cioè il processo che ha determinato la non rilevazione è completamente indipendente dal valore mancante ma dipende da altre variabili. Pertanto si è deciso semplicemente di rimpiazzare con 0 i *missing values*. In questo modo i valori mancanti rimanenti sono 173, quindi il 12% delle istanze per la variabile *"prezzo_numerico"*. Tuttavia in questo caso la questione è di ben altra natura perchè il processo che ha determinato la non rilevazione è completamente indipendente dal valore mancante e da qualsiasi altra variabile disponibile e soprattuto è aleatorio, quindi si ha bisogno di un approccio differente: quello dell'imputazione attraverso metodi di regressione.

2.1 Regressione per Imputazione

Come specificato siamo nel caso in cui la probabilità che un certo valore per una data variabile sia missing è indipendente da qualsiasi altro valore per qualsiasi altra variabile. Ovvero i valori missing values sono distribuiti aleatoriamente. Nella teoria della gestione dei valori mancanti si parla di due diversi protocolli d'azione: uno passivo e uno attivo. La strategia passiva prevede di ignorare i missing values e procedere all'analisi sui dati presenti. Si eliminano le osservazioni con i dati mancanti e questa soluzione è quella ottimale se i dati mancanti interessano poche unità, quindi se ci si trova in un contesto a basse proporzioni di missing (5% o meno). Non è questo il caso, questa procedura porterebbe ad una perdita sostanziale di osservazioni e informazioni. La strategia attiva invece ha come obiettivo quello di sostituire ciascun valore mancante con uno plausibile, stimato sulla base dei valori validi delle altre variabili complete. Questa è la definizione di imputazione, una metodologia tanto potente quanto pericolosa, infatti si dovrebbe considerare in modo molto cauto l'uso di quest'ultima, per via del suo potenziale impatto, talora molto forte, sull'analisi dei dati. Vi sono diversi modi per imputare, di sicuro quello più sofisticato è quello tramite la regressione. La stima del dato mancante può essere ottenuta stimando un modello di regressione in cui la variabile con i casi mancanti è usata come variabile dipendente e le altre sono utilizzate come variabili esplicative oppure utilizzando algoritmi

di machine learning. Il modello è stimato sulla base dei casi completi ed è impiegato per prevedere i casi mancanti. Questo metodo è sicuramente il più raffinato; può però soffrire dei problemi legati alla qualità del modello di regressione utilizzato. Sono state tre le metodologie utilizzate per regredire sui dati, tutte appartenenti alla sfera del machine learning: SVM, Processi Gaussiani e KNN.

Prima di spiegare e mostrare i risultati di questi tre algoritmi è stata svolta un'ulteriore fase di lavorazione dei dati. É stata creata una nuova variabile, dummy, che assume valore 1 se il calciatore milita nel campionato inglese, 0 altrimenti. Il senso di questa scelta è chiaro dalla figura 1, in cui è evidente che il valore di mercato dei giocatori in Premier League è tendenzialmente maggiore rispetto a quello dei calciatori che militano negli altri quattro campionati. Le ragioni di questa discrepanza risiedono non solo nella qualità dei calciatori effettivamente maggiore presente in Inghilterra, ma anche nelle questioni di popolarità della lega e quindi di disponibilità economica derivante dai ricavi dei diritti di streaming delle partite.

Successivamente si è diviso il dataset in due: una parte con i dati completi, l'altra composta solo dai record con missing values. Questa seconda parte sarà trattata come test set per gli algoritmi che si andranno ad allenare. La dimensionalità del dataset è piuttosto elevata e per far sì che gli algoritmi possano lavorare al meglio è necessario ridurla. Ci sono diversi approcci a questo problema, ma in questo caso si è deciso per una PCA sui dati standardizzati. Le ragioni di questa scelta si possono ricondurre al fatto che la PCA è il metodo più immediato ed efficace per ottenere una riduzione della dimensionalità; inoltre il suo più importante svantaggio, cioè la perdità di interpretabilità, non risulta essere un problema in questo caso.

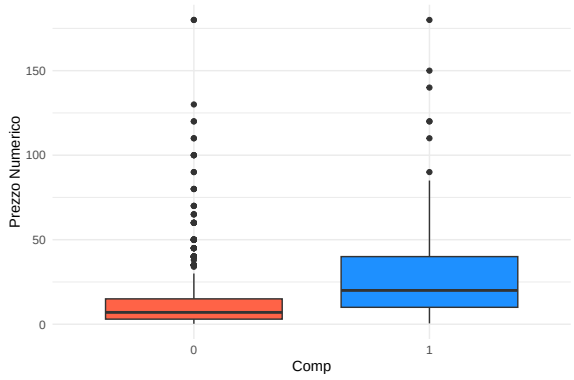


Figure 1: Boxplot del Valore di Mercato se un calciatore milita in English Premier League oppure in un'altra lega

Si è deciso di considerare le prime dieci componenti principali che spiegano circa l'84% circa della varianza. Infine si è suddiviso il dataset composto dalle dieci componenti in Training set corrispondente al 80% delle osservazioni e in Validation set.

2.1.1 Support Vector Machine per Regressione

Il Support Vector Machine è un algoritmo che nasce per risolvere problemi di classificazione. L'idea alla base è però stata allargata anche ai problemi di regressione. L'obiettivo è quello di trovare l'iperpiano che racchiuda la maggior parte delle osservazioni all'interno delle sue bande di tolleranza. É chiaro come questa sia una richiesta ambiziosa da soddisfare, di certo vi sarà qualche osservazione che cadrà fuori dai margini stimati. Pertanto sono introdotte le *slack variables* che permettono ad alcuni punti di violare il margine pur mantenendo la robustezza del modello. L'obiettivo è minimizzare il numero e l'entità di queste violazioni, tant'è che viene introdotto l'iperparametro **c** per gestire il trade-off tra la complessità del modello e il livello di flessibilità delle deviazioni dalle bande di tolleranza. Un'altra questione da tenere in conto è la possibilità che i dati non abbiano un pattern lineare. Ciò porta a

tenere in considerazione l'ipotesi di lavorare con una funzione kernel che permetta di trasformare i dati per rendere possibile la separazione lineare di quest'ultimi.

Attraverso un algoritmo di *Automated Machine Learning* si sono confrontati due iperpiani di separazione differenti: uno *lineare* e il secondo che sfrutta una funzione kernel di tipo *Gaussiana*. Gli iperparametri necessari sono **c** il costo di avere osservazioni fuori dal margine, γ che è esclusivo del kernel Gaussiano e che controlla l'ampiezza dell'influenza dei singoli punti di addestramento sul *decision boundary* ed ϵ che controlla l'ampiezza delle bande di tolleranza. I primi due sono stati fatti variare in un range che parte da 10^{-2} e arriva a 10^2, mentre il terzo prende valori nell'intervallo $[0.001, 0.999]$.

L'ottimizzazione è effettuata attraverso una *10fold-cross validation* come metodologia di ricampionamento per valutare le diverse combinazioni di valori degli iperparametri. Inoltre si è scelto come misura per valutare la performance del modello l'RMSE. I risultati mostrano che il migliore è il modello caratterizzato da un iperpiano di separazione non lineare, un valore di **c** pari a 2.95 e di γ uguale a 0.015. Come **c** approccia lo 0 la tolleranza tende a 0 facendo sì che la previsione del Support Vector Regression coincida con il caso più semplice: quello che non considera le *slack variables*. In questo caso **c** è relativamente piccolo rispetto alla scala di valori che può assumere quindi si può concludere che il modello ottimo preferisce essere meno flessibile, quindi accrescere il bias ma ridurre la varianza. Anche γ assume un valore basso rispetto all'intervallo di valori possibile e ciò significa che ogni punto di addestramento ha un'influenza più ampia, il che tende a produrre *decision boundaries* più lisci e meno complessi. Il numero di Support Vectors individuati dal modello è 566, il 56% circa delle osservazioni del training set. Questo è un forte campanello di allarme, infatti un numero eccessivo di Support Vectors spesso indica che il modello sta cercando di adattarsi troppo ai dati di addestramento, inclusi rumore e outlier. Questo adattamento eccessivo può portare a un modello che non generalizza bene, risultando in prestazioni scarse sul test set, e questa è la definizione di *overfitting*. Infine è stato calcolato l'errore di Generalizzazione che è mostrato nella tabella 2, che risulta pari a 16.642.

2.1.2 Processi Gaussiani per Regressione

I Processi Gaussiani possono essere interpretati come metologie per definire distribuzioni su un insieme di funzioni. Ciò li permette di rappresentare per la regressione un approccio probabilistico non parametrico per modellare una funzione sconosciuta. Una caratteristica fondamentale è la flessibilità nella capacità di poter adattarsi a contesti di regressione non lineare. Si sono considerati i GP come algoritmo di ML per la regressione perchè si trattano di modelli probabilistici che forniscono una distribuzione di probabilità per le predizioni, consentendo di quantificare l'incertezza, al contrario di SVM e KNN che, invece, producono predizioni deterministiche.

La scelta del kernel è fondamentale perchè definisce la struttura della funzione di covarianza, determinando la correlazione tra i punti nel dominio. La teoria afferma che si debba preferire una funzione kernel che rappresenti la conoscenza a priori della struttura dei dati, nel caso in questione, però, si è optato per un kernel ***Squared Exponential*** perchè è quello che garantisce maggior *"smoothness"*.

Tramite *Kriging*, una tecnica di interpolazione spaziale ampiamente utilizzata in geostatistica, si può costruire un modello di processo gaussiano per la regressione. In questo modo si arriva al risultato ottimo che, come mostrato nella tabella 2, performa un errore di Generalizzazione pari a 19.387.

2.1.3 Knn

Il K-Nearest Neighbors è un algoritmo di Machine Learning non parametrico. É stato scelto perchè è tanto semplice e intuitivo quanto efficiente. Il suo principio fondamentale è che i punti di dati simili (vicini) tendono ad avere valori di output simili. Per effettuare una predizione, l'algoritmo identifica i k punti di addestramento più vicini al nuovo punto di input e calcola la media dei valori di output di questi k vicini. Non ha inoltre bisogno di particolari assunzioni sui dati: nei fatti si adatta naturalmente a quest'ultimi.

La scelta del parametro k è cruciale: un k troppo piccolo può rendere il modello suscettibile al rumore (overfitting), mentre un k troppo grande può far perdere le specificità locali dei dati (underfitting). Si è deciso quindi di calcolare l'errore di generalizzazione per ogni modello stimato con un numero diverso

di vicini da 1 a 20. Si è scelto questo intervallo di valori per k in modo arbitrario, ovviamente tenendo in conto le ragioni poco prima descritte, cioè il cercare di mantenere un equilibrio tra l'*overfitting* e l'*underfitting*. Il k che minimizza l'errore è pari a otto, cioè il modello ottimale è un 8-NN, che è caratterizzato da un valore di errore di Generalizzazione pari a 18.128.

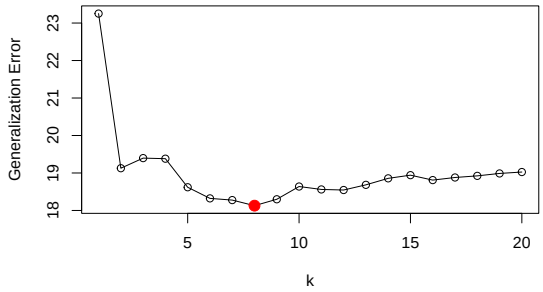


Figure 2: KNN-Generalization Error al variare di k

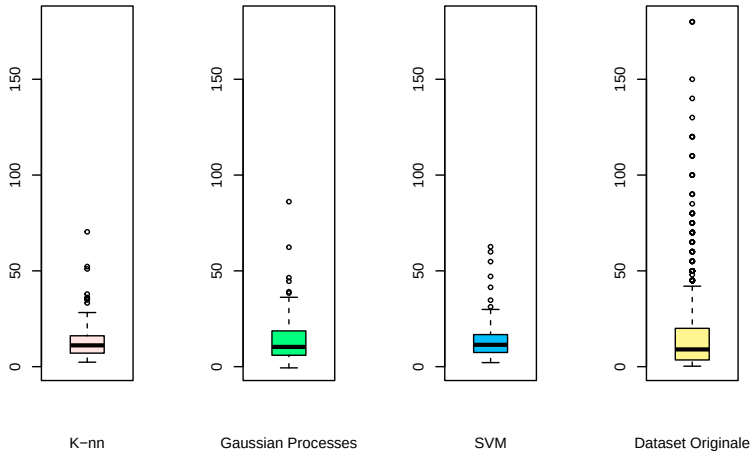


Figure 3: Boxplot Valori di Mercato suddivisi per algoritmo

Algoritmo	Min	1st Qu.	Mediana	Media	3rd Qu.	Max
KNN	2.331	7.062	11.125	13.306	16.125	70.375
GP	-0.631	5.954	10.308	13.481	18.666	86.127
SVR	2.115	7.395	11.401	13.632	16.742	62.594
Dataset Orig.	0.25	3.50	9.00	16.77	20.00	180.00

Table 1: Statistiche Descrittive algoritmi e dataset originale

2.1.4 Conclusioni Imputazione

Si mettono a confronto i risultati ottenuti tramite i tre algoritmi, calcolando le rispettive previsioni sul Test set, cioè su quelle osservazioni che in precedenza avevamo lasciato da parte con valore mancante in corrispondenza della variabile *prezzo_numerico*. Una prima comparazione a livello qualitativo avviene attraverso il confronto tra le statistiche descrittive delle previsioni mostrate nella figura 3 e nella tabella 1. Si devono subito escludere i Processi Gaussiani i quali hanno almeno una previsione negativa essendo il loro minimo pari a -0.6305; un valore impossibile. Per quanto riguarda invece SVM e KNN entrambi hanno statistiche descrittive simili a quelle del dataset originale. Per quanto riguarda l'errore di Generalizzazione si vede dalla tabella 2 che l'algoritmo di Support Vector Regression è migliore rispetto all'8-NN. Si ricorda però, come detto in 2.1.1, che il numero di Support Vectors individuati è elevato e ciò può essere sintomo di *overfitting*. Per questo motivo è ragionevole scegliere come algoritmo per l'imputazione il K-NN. Le previsioni che si sono ricavate sono ovviamente limitate, non tanto per la non efficienza degli algoritmi quanto per l'approssimazione del modo in cui ci si è approcciati. Infatti è giusto far presente che il valore di mercato di un calciatore è una quantità alla cui determinazione compartecipano cause di varia natura; alcune addirittura estranee ai fatti sul campo. É questa la ragione per cui, come si vede nella figura 3, rispetto ai valori previsti con gli algoritmi il dataset originale ha molti più valori maggiori del terzo quartile. Alcuni sono molto lontani tant'è che possono essere considerati outliers.

Algoritmo	Errore di Generalizzazione
SVR	16.642
Processi Gaussiani	19.387
KNN	18.128

Table 2: Errori di previsione degli algoritmi ML

3 Classificazione

Nel calcio moderno, la versatilità dei giocatori e la loro capacità di adattarsi a diverse posizioni è diventata cruciale. Tuttavia, ogni posizione in campo richiede un set specifico di abilità e caratteristiche. Tradizionalmente, la valutazione delle posizioni dei giocatori è stata fatta basandosi sull'osservazione diretta e sull'esperienza degli allenatori. Con l'avvento dei big data e delle tecniche di machine learning, è possibile adottare un approccio più quantitativo e sistematico per la classificazione delle posizioni. Durante l'analisi di classificazione si è cercato di sfruttare le statistiche estratte dai siti citati [2][5] per costruire il miglior modello di classificazione con lo scopo di prevedere la posizione in campo dei diversi calciatori. Il dataset propone la distinzione delle classi in 4 macro categorie principali: *Goalkeeper*, *Defender*, *Midfielder* e *Forward*. La rappresentazione delle numerosità campionarie delle etichette relative alle posizioni all'interno dell'intero dataset è espressa in tabella 3.

Goalkeeper	Defender	Midfielder	Forward
110	573	436	299

Table 3: Frequenze assolute delle classi nell'intero dataset

Come visibile le classi sono piuttosto bilanciate, nonostante vi sia un numero di portieri leggermente inferiore rispetto ai calciatori presenti nei restanti tre ruoli. Non è stato ritenuto opportuno intervenire con algoritmi di bilanciamento delle classi, nè prima della partizione del dataset e nemmeno successivamente. Come vedremo in seguito non ci saranno problemi riguardanti la classificazione della classe *Goalkeeper*.

I modelli utilizzati per lo studio del task di classificazione sono principalmente due: **Support Vector Machine** e **Random Forest**.

Le trasformazioni dei dati sfruttate per questo task sono differenti in base all'algoritmo utilizzato e alle ipotesi che ne derivano. Per quanto riguarda l'algoritmo di **SVM** sono stati adoperate le stesse trasformazioni della sezione precedente 2.1, tuttavia si è ritenuto opportuno includere anche la variabile prezzo numerico all'interno dell'analisi in componenti principali sui dati standardizzati per la riduzione di dimensionalità e la selezione della feature più opportune. Anche in questo caso i dati usati sono stati standardizzati poichè l'algoritmo **SVM** sfrutta delle misure di distanza come la distanza euclidea.

Si è inoltre considerato di mantenere un numero di componenti principali tali che spiegassero almeno l'85% della variabilità del dataset.

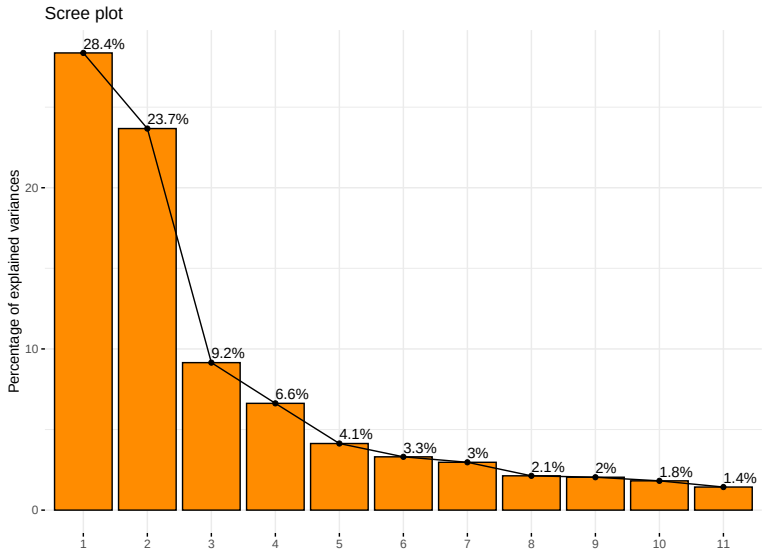


Figure 4: Varianza spiegata dalle prime 11 componenti della PCA

Per quanto riguarda l'algoritmo **Random Forest**, invece, si è deciso di utilizzare il dataset completo di partenza, senza alcuna trasformazione, per le motivazioni che verranno rese esplicite in seguito (sezione 3.2).

Si è quindi deciso, come visibile in figura 4, di considerare soltanto le prime 11 componenti dell'analisi PCA. Il set di dati utile al nostro task di classificazione ora è quindi composto da un set di variabili qualitative che descrivono i calciatori (tra cui la variabile target) e tra queste sono presenti le variabili:

- *Season End Year*: Indica l'annata calcistica considerata (2024 in questo caso);
- *Squad*: Squadra in cui gioca il giocatore;
- *Comp*: Il campionato di origine della squadra in cui milita il calciatore: è una variabile fattoriale a 5 modalità: *Premier League*, *Serie A*, *Ligue One*, *Bundesliga*;
- *Player*: Nome del calciatore;
- *Nation* : Nazionalità;
- *Pos*: Variabile target;
- *Born*: Anno di nascita;

Queste variabili qualitative saranno poi utili ai fini di individuare e diagnosticare eventuali errori di previsione del modello in esame e visualizzare le 'vere' statistiche associate al calciatore erroneamente predetto per andare a fare un'eventuale interpretazione della mancata corretta classificazione.

Prima di implementare i due modelli si è diviso il dataset di partenza in training e test set, il primo utilizzato per allenare gli algoritmi e il secondo per verificarne l'accuratezza e per la validazione degli iperparametri utilizzati. In particolare il training set è stato creato considerando il 75% delle istanze, mentre il test con le restanti. Per entrambe le trasformazioni sono state utilizzate le stesse istanze per la divisione in training e test set.

3.1 Support Vector Machine

Un primo strumento per risolvere il task di classificazione del nostro problema è l'algoritmo di Support Vector Machine per variabile target multiclasse. Questo algoritmo permette di confrontare diversi esempi di *decision boundary* in modo da trovare la frontiera decisionale più adatta alla complessità dei dati in esame. Durante l'implementazione dell'algoritmo è da tenere in considerazione il costo computazionale dato da frontiere di decisioni più complesse rispetto a *decision boundary* di tipo lineare. Sono stati presi in considerazione tre diversi tipi di iperpiani di separazione, non conoscendo a priori la complessità dei dati, e messi a confronto attraverso un algoritmo di *Automated Machine Learning*. In particolare sono stati confrontati 3 iperpiani di separazione differenti: uno *lineare*, mentre altri due non lineari che sfruttano una funzione kernel di tipo *Gaussiana* e di tipo *Polinomiale*. Gli iperparametri da valutare sono essenzialmente quattro:

1. **c**: viene fatto variare in un range di valori che va da 0.01 a 100 e stabilisce il trade-off tra la massimizzazione del margine di separazione e la minimizzazione degli errori di classificazione;
2. γ: valido solo per le funzioni Kernel polinomiali e gaussiana e viene fatto variare in un range di valori che va da 0.01 a 100;
3. **degree**: ossia il grado della funzione polinomiale e viene fatto variare da 1 a 10 (oltre il 10 la complessità dei dati sarebbe considerata troppo elevata e il modello non sarebbe più in grado di generalizzare). Questo iperparametro è valido soltanto per il Kernel polinomiale.

L'ottimizzazione, effettuata sul training set attraverso l'utilizzo di una *10fold-cross validation*, mostra che l'iperpiano ottimo separazione è di tipo lineare con un parametro c pari a 0.239. Questo risultato afferma che i dati presi in considerazione, a seguito delle opportune trasformazioni, sono strutturati in maniera piuttosto semplice per il task di classificazione tanto da non necessitare dell'utilizzo di *decision boundary* complessi. Il valore di c è relativamente piccolo se si tiene in considerazione la griglia prestabilita di possibili iperparametri. Ciò significa che il modello permetterà un margine di separazione più ampio e sarà più tollerante verso gli errori di classificazione. In altre parole, il modello preferisce avere un margine più ampio con alcuni errori di classificazione piuttosto che cercare di classificare correttamente tutti i campioni di training. Un valore di c basso, come in questo caso, tende a ridurre l'overfitting, poiché il modello è più flessibile e generalizza al meglio i dati di test. Tuttavia un problema a cui andremo incontro è l'*underfitting*, ossia il modello generalizza troppo i dati e le osservazioni più complesse da prevedere verranno classificate erroneamente.

A seguito dell'ottimizzazione degli iperparametri l'algoritmo è stato valutato per fornire una stima dell'errore empirico e dell'errore di generalizzazione attraverso un *10-fold validation schema*, dove il training set viene suddiviso per 10 volte in training e validation (in maniera diversa in modo da evitare overlap). Il modello con parametri ottimali viene stimato sul nuovo training e valutato nel validation. Grazie a questa tecnica si è in grado di valutare l'overfitting.

La metrica utilizzata per la valutazione del modello è l'*accuracy*. Come ben visibile da tabella 4 il modello scelto non porta all'overfitting. E' possibile quindi effettuare previsione sul test set estratto in precedenza. In tabella 4 sono rappresentati i risultati relativi ai diversi errori generati nel fare previsione.

L'accuracy non è l'unica metrica per valutare la previsione del modello. Osservando la matrice di confusione rappresentata in tabella 5 è possibile osservare come la classe dei portieri sia perfettamente prevista.

Altre misure come la *Sensitivity* e *Specificity* sono rappresentate in tabella 6:

Errore	Valore	Accuracy
Empirico	0.1211	0.8788732
Generalizzazione	0.123	0.877
Test/Previsione	0.1416	0.8584

Table 4: Errori di previsione del modello SVM

	Valori Reali			
Valori Previsti	GK	DF	MF	FW
GK	27	0	0	0
DF	0	127	15	0
MF	0	16	80	5
FW	0	0	14	69

Table 5: Matrice di confusione algoritmo Support Vector Machine

Si può notare come entrambe le misure siano pari a 1 per la classe dei portieri dato che questi vengono tutti predetti perfettamente. Si evidenziano, invece, valori piuttosto bassi di Sensitivity per quanto riguarda le posizioni dei centrocampisti. Questo porta ad affermare che il modello si comporta in maniera non ottimale quando si tratta di classificare la posizione dei calciatori che giocano nella parte centrale del rettangolo verde. Questo risultato può essere ricondotto a quanto detto in precedenza riguardo all'underfitting del modello. Il modello implementato è molto semplice e classifica in maniera corretta tutte quelle osservazioni che si trovano lontane dagli iperpiani lineari di separazione, mentre in maniera errata le osservazioni che si trovano proprio sul confine. Essendo il ruolo del centrocampista un ruolo molto duttile ed adattibile a qualsiasi zona di campo, in passato si è visto spesso centrocampisti assumere per alcune partite il ruolo di attaccante o difensore, avere quindi statistiche molto simili ad altri ruoli, porterà l'eventuale punto d-dimensionale a posizionarsi molto vicino alle *decision boundary* stabilite dal modello e ad essere classificato erroneamente.

Attraverso un'analisi grafica più approfondita sono stati valutati quali sono i giocatori che sono stati classificati in modo errato e verificato se le ipotesi viste in precedenza siano corrette. Concentrandosi sui centrocampisti, in figura 5, è possibile vedere quanto detto in precedenza. In figura 5 è infatti rappresentato un grafico a dispersione delle prime due componenti principali (che spiegano circa il 50% della variabilità totale delle feature) e vengono colorati in rosso tutti centrocampisti del test set che sono stati classificati erroneamente dal modello.

	GK	DF	MF	FW
Sensitivity	1.000	0.8881	0.7339	0.9459
Specificity	1.000	0.9286	0.9498	0.9139

Table 6: Specitivity e Sensitivity del modello SVM

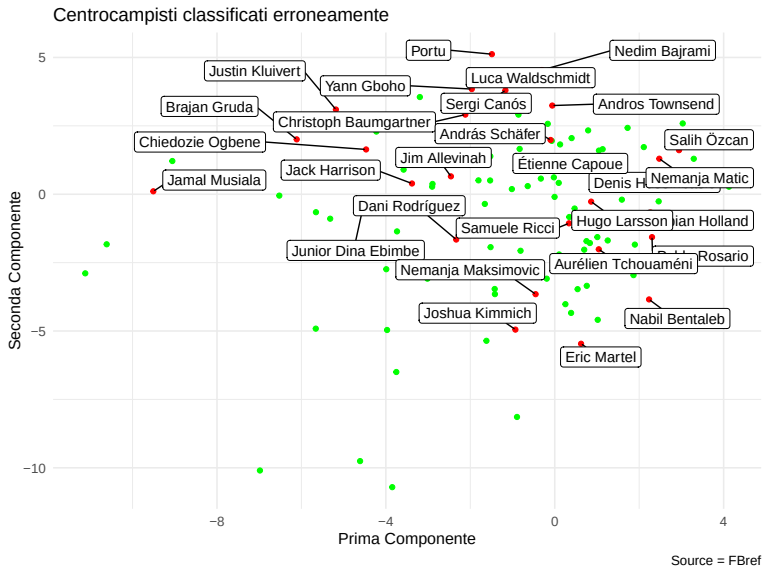


Figure 5: Centrocampisti classificati erroneamente da SVM

I nomi dei calciatori associati a queste osservazioni non fanno altro che confermare le ipotesi fatte in precedenza. Si prende come esempio il calciatore Jamal Musiala, centrocampista del Bayern Monaco, che ha uno stile di gioco molto offensivo e questo aspetto viene evidenziato dalle statistiche a lui associate. Molto probabilmente il suo punto in uno spazio 11-dimensionale è vicino alla categoria degli attaccanti tanto da non esserne escluso quando il modello applica il metodo 'one vs other' delineando un iperpiano di separazione che lo include. Per questo motivo il calciatore viene classificato erroneamente come attaccante. Casi completamente opposti possono essere invece rappresentati dai centrocampisti Aurélien Tchouaméni e Samuele Ricci che possedendo qualità e statistiche più difensive venendo erroneamente classificati come difensori.

Essendo il ruolo del centrocampista il ruolo più duttile e versatile, i risultati ottenuti dal modello e la sua sensitivity erano in una certa misura ciò che ci si poteva attendere.

3.2 Random Forest

La scelta del modello Random Forest per classificare le posizioni dei calciatori utilizzando le statistiche di FBref [2] è motivata da diversi fattori tecnici e pratici che rendono questo algoritmo particolarmente adatto per questo tipo di analisi. Il Random Forest è noto per la sua robustezza e capacità di gestire dataset complessi con molte caratteristiche. Nello specifico, il dataset analizzato fornisce una una vasta gamma di dati stagionali dei singoli calciatori, tra cui tiri, passaggi, gol, salvataggi e altro ancora. L'algoritmo è in grado di gestire questo alto numero di variabili e di identificare le caratteristiche più rilevanti per distinguere le diverse posizioni in campo. Per questo motivo durante l'implementazione si è deciso di utilizzare il dataset completo e non traformato attraverso l'analisi in componenti principali. Una seconda motivazione è data dal grado di interpretabilità che questo algoritmo mantiene. È possibile, infatti, calcolare l'importanza delle varie caratteristiche (*feature importance*) e identificare quali statistiche siano più influenti nella classificazione delle posizioni. Questo è utile per gli allenatori e gli analisti, che possono ottenere intuizioni significative su quali attributi sono cruciali per diversi ruoli in campo. Un' ultima motivazione per cui si è scelto di utilizzare il dataset completo a discapito del set di dati trasformato attraverso PCA è data dai risultati evidenziati dal modello **SVM** stimato al paragrafo 3.1. Il modello al paragrafo 3.1 afferma che i dati trasformati sono semplici e linearmente separabili. L'algoritmo di Random Forest fornisce dei *decision boundary* complessi e non lineari, adatti a set di dati non linearmente separabili come i nostri dati di partenza. I dati, inoltre, non vengono scalati dato che il Random Forest non si basa su metriche di distanza a differenza del modello **SVM**. Per allenare

il modello viene utilizzato, anche in questo caso, un *10-Fold Validation Schema* in modo da riuscere ad ottenere una stima dell'errore di generalizzazione e valutare un eventuale overfitting.

In tabella 7 sono quindi rappresentati i risultati dell'addestramento del modello di Random Forest e delle suo previsioni sul test set. Anche in questo caso la metrica utilizzata è l'accuracy. Come ben visibile non si è in presenza di overfitting, dato un errore di generalizzazione molto vicino all'errore empirico e un errore di previsione relativamente più piccolo, addirittura più piccolo dell'errore empirico.

Errore	Valore	Accuracy
Empirico	0.1099	0.89014
Generalizzazione	0.1088	0.8912
Test/Previsione	0.09348	0.9065

Table 7: Errori di previsione del modello Random Forest

E' facile osservare come l'errore di generalizzazione sia inferiore all'errore empirico, fatto che teoricamente non dovrebbe accadere. Tuttavia se si osservano i valori assunti durante il *10-validation schema* si nota una vasta variabilità di questo errore. Inoltre, essendo il campione relativamente piccolo porta a questa situazione anomala. Tuttavia siccome errore empirico ed errore di generalizzazione sono molto vicini tra loro, questo risultato, contrario alle leggi teoriche, non è considerato statisticamente significativo.

In tabella 8 viene invece rappresentata la matrice di confusione.

	Valori Reali			
Valori Previsti	GK	DF	MF	FW
GK	27	0	0	0
DF	0	133	5	0
MF	0	10	90	4
FW	0	0	14	70

Table 8: Matrice di confusione algoritmo Random Forest

L'algoritmo fornisce inoltre una metrica per valutare l'importanza delle feature utilizzate nel task di classificazione delle posizioni dei calciatori. In figura 6 sono rappresentate le 15 variabili più importanti secondo la metrica di Gini. La metrica di Gini è un metodo che misura quanto ciascuna variabile contribuisce a ridurre l'impurità nelle suddivisioni dei nodi degli alberi della foresta.

Come visibile le variabili che discretizzano al meglio le posizioni secondo la metrica di Gini sono *Crl* e *Def.3rd Touches*. La statistica CLR (Completion %, Launch %, Reception %) in FBRef è una metrica utilizzata per analizzare le performance dei giocatori nel calcio, specialmente per i giocatori che operano in posizioni avanzate come centrocampisti e attaccanti ed è composta da tre fattori: la percentuale di passaggi riusciti, la percentuale di lanci riusciti e la percentuale di ricezioni riuscite. Def.3rd Touches indica, invece, semplicemente i tocchi dei calciatori nella trequarti difensiva. Come ben visibile le due feature per importanza nella discretizzazione della variabile posizione riguardano specificamente ruoli offensivi, per la prima, e ruoli difensivi per la seconda. Se si osserva la terza variabile per importanza, *Sh per 90*, ossia i tiri in porta ogni novanta minuti, è verosimile pensare che ad un attaccante siano attribuiti valori più elevati rispetto ad un difensore o un centrocampista, ben che meno di un portiere.

Osservando gli errori di classficazione del Random Forest è possibile notare come si sia riscontrato lo stesso problema del modello al paragrafo 3.1. Prendendo in considerazione questa volta i difensori missclassificati è possibile notare come questi siano tutti giocatori offensivi, che giocano sulle fasce come terzini o in un sistema di gioco molto offensivo. Un esempio è Raphaël Guerreiro, difensore centrale del Borussia Dortmund che nell'ultima stagione ha giocato in una posizione media molto avanzata rispetto agli standard di un classico difensore centrale. Diversamente accade, per esempio, per giocatori come Trent Alexander-Arnold o Nadir Zortea, che essendo giocatori a tutta fascia con una propensione

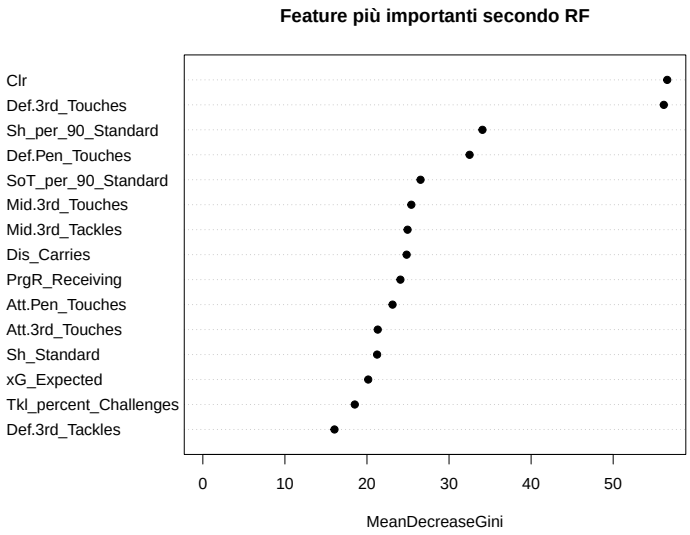


Figure 6: 15 feature più importanti secondo Random Forest per la classificazione delle posizioni

all'attacco le loro statistiche saranno molto simili a quelle di un centrocampista e probabilmente per questo motivo sono stati classificati erroneamente.

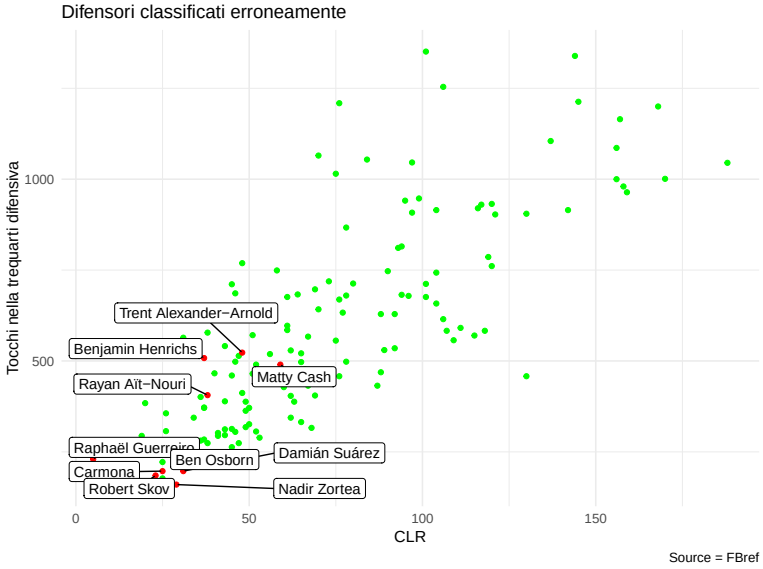


Figure 7: Difensori classificati erroneamente dal Random Forest

Analizzando approssivamente, attraverso la rappresentazione in figura 8, le sei statistiche più importanti stabilite dall'algoritmo, di uno dei difensori classificati erroneamente dal Random Forest, come Trent Alexander-Arnold, è possibile notare come queste siano molto più simili alle statistiche medie di un centrocampista (rappresentate in rosso) rispetto a quelle dei difensori (rappresentate in giallo). Si può osservare come al giocatore corrispondano valori bassi rispetto alle variabili *Def.3rd* e *Clr* conformi con le medie dei centrocampisti, mentre per i difensori si riscontrano valori medi più elevati. Il contrario invece si nota per la variabile caratterizzante i tiri ogni 90 minuti, *Sh per 90*.

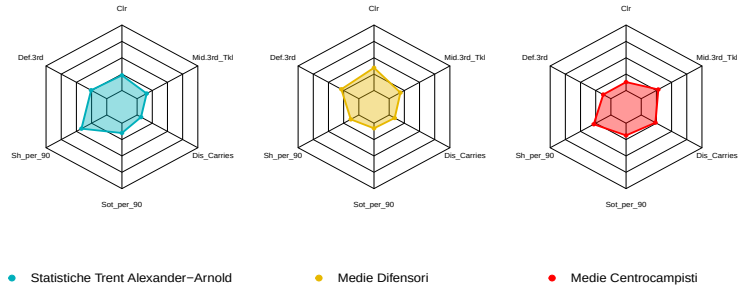


Figure 8: Confronto tra le statistiche del difensore Trent Alexander-Arnold (a sinistra) e le rispettive medie di difensori (in centro) e centrocampisti (a destra) per le 6 variabili più importanti

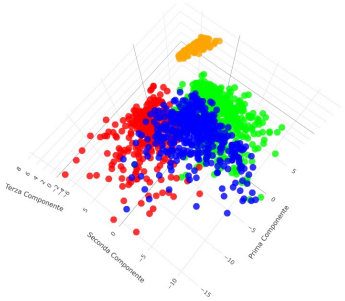
4 Clustering

In questa sezione verranno affrontati due algoritmi per il clustering con lo scopo identificare pattern e strutture nelle statistiche riferite al gioco del calcio[2][5]. L'analisi viene eseguita sulla trasformazione del dataset già descritta nella sezione 3, ossia sulle prime 11 componenti principali calcolate sui dati standardizzati. Vengono utilizzati i dati standardizzati poichè gli algoritmi che verranno studiati si basano su metriche di distanza e sfruttano assunzioni che prediligono dati che abbiano la stessa scala e varino nello stesso range di valori. Inoltre, riducendo la variabilità dei dati, si riducono i tempi computazionali dell'algoritmo per arrivare a convergenza, poichè le variabili standardizzate hanno varianze simili tra loro e quindi più facilmente comparabili. In figura 9 è rappresentato uno scatterplot tridimensionale delle prime tre componenti differenziato per la variabile posizione. Come facilmente visibile è presente un pattern giallo di punti complementamente distanziato dalla massa di punti presente al centro dell'iperpiano tridimensionale. Questo pattern ben distinto porta a considerare una maggiore investigazione delle relazioni tra i dati attraverso l'implementazione di due algoritmi di clustering: **k-means clustering** e **Gaussian Mixture Model**. Durante l'implementazione si è supposto di non conoscere le etichette relative alle posizioni dei calciatori e di trovarsi un contesto non supervisionato. Tuttavia una volta sviluppati i metodi si è deciso di far riferimento alle posizioni e osservare come questi legavano le diverse etichette.

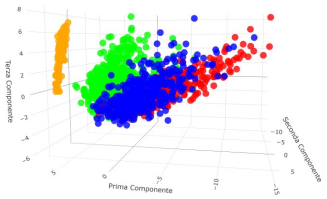
4.1 K-Means Clustering

Il primo algoritmo implementato è il **K-Means clustering**. L'algoritmo porta alla creazione di cluster di forma ipersferica. Il k ottimale di gruppi viene scelto attraverso il metodo del gomito (*Elbow Method*) e l'utilizzo della *Within Cluster Sum of Squares*(WCSS). Questa è una tecnica di validazione interna comune per il k-means e permette di trovare il numero ottimale di cluster k in modo tale che l'aggiunta di ulteriori cluster non migliora significativamente la varianza all'interno di ciascun gruppo. L' obiettivo è quindi quello di costruire cluster il più eterogenei tra loro, ma il più omogenei al loro interno. Il K ottimale ottenuto è pari a 4. In questo caso fortunato si è derivato un numero di cluster esattamente pari al numero delle posizioni dei calciatori. Stimando il modello si sono ottenute le numerosità espresse in tabella 9.

Si evidenza subito come nel cluster 1 sia presente un numero di istanze pari a 110, come l'esatto numero dei portieri, espresso in tabella 3. Osservando le istanze appartenenti al cluster 1 è infatti ben



(a) *Scatterplot 3D visto dall'alto*



(b) *Scatterplot 3D visto frontalmente*

Figure 9: Scatterplot 3D delle prime tre componenti differenziato per posizione

Cluster	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Numerosità	110	368	606	334

Table 9: Numerosità dei cluster ottenuti tramite K means clustering

visibile come questi siano tutti e 110 dei portieri. Il cluster in questione è proprio la nuvola gialla che si era distinta negli scatterplot precedenti (figura 9).

I risultati del clustering sono rappresentati in figura 10 attraverso uno scatterplot bidimensionale delle prime due componenti principali. Come ben visibile il pattern arancione di punti caratterizzato dai portieri, è distinto anche all'interno dei cluster forniti dall'algoritmo. Una distinzione non così netta invece, viene riscontrata negli altri ruoli. Si può infatti notare come la classe dei centrocampisti (rappresentata in blu nella proiezione a due dimensioni) sia posizionata internamente rispetto alle altre due classi, ma sparsa per tutto il piano cartesiano. Questa sparsità dei centrocampisti non viene tuttavia colta dal K-means cluster. Questo è dato dalle ipotesi dell'algoritmo che associano un'istanza al cluster che ha il centroide più vicino in termini di distanza euclidea e quindi tenderà a determinare ipersfere di punti vicini tra loro. Sarebbe impensabile, date le ipotesi dell'algoritmo, riscontrare una situazione come quella descritta dal grafico di dispersione nella figura 10 in alto, dove i punti blu si trovano dispersi per tutto il piano cartesiano. Questa dispersione dei centrocampisti concerne con quanto spiegato nella sezione 3 relativa alla classificazione. Ossia che il ruolo del centrocampista, essendo un ruolo così duttile e facilmente adattabile a diverse situazioni di gioco, può portare istanze di statistiche di tutte le specie dalle più offensive (molto vicini agli attaccanti in termini di distanza euclidea) e quindi raggruppati nell' insieme composto prevalentemente da attaccanti, oppure con pattern molto più simili a quelli dei difensori.

Osservando le percentuali delle classi di posizione all'interno di ciascun singolo gruppo (espresse in figura 11), è possibile notare come, ad esempio, nel primo cluster siano presenti per il 72% attaccanti, per il 26% centrocampisti e per il 2% difensori. Studiando nomi e statistiche relative a questi calciatori si nota:

- Tra i difensori inseriti nel cluster 1 sono presenti giocatori come Federico Dimarco, difensore esterno dell'Inter e Jeremie Frimpong; giocatori molto offensivi, reduci da una stagione ricca di assist e gol, oltre che ad uno stile di gioco propenso all'attacco. Analizzando i dati a loro associati è possibile notare come questi siano molto più vicini a ruoli offensivi come attaccanti e centrocampisti rispetto che alle statistiche medie dei classici difensori.

- Se si osservano invece i centrocampisti inseriti dall'algoritmo all'interno del cluster 1 è possibile notare nomi come Kevin De Bruyne o il già sopracitato, durante la classificazione, Jamal Musiala, ossia quei centrocampisti con statistiche, gol e prestazioni da veri e propri attaccanti.

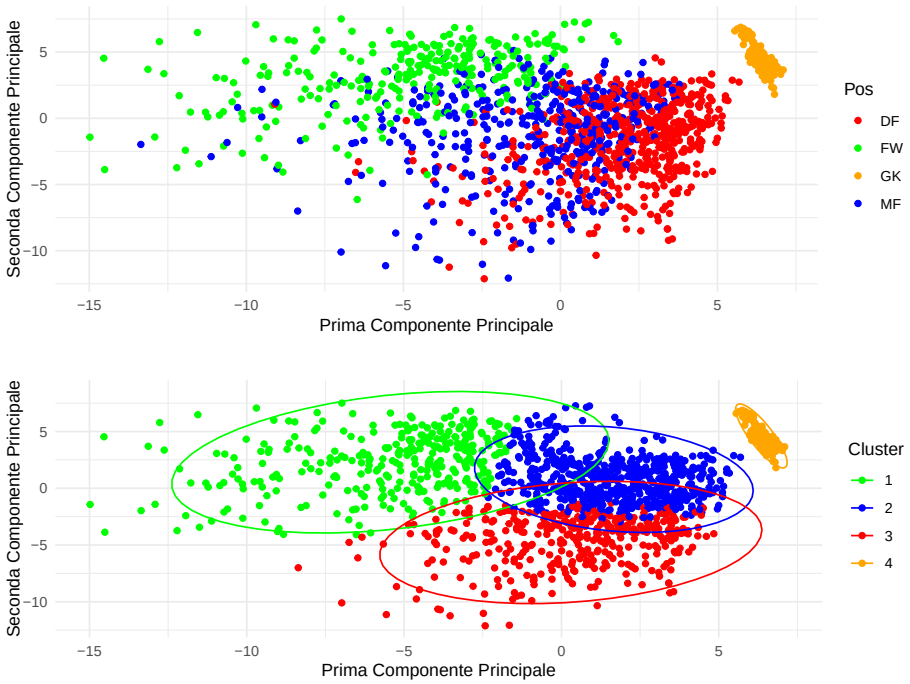


Figure 10: Valori reali delle posizioni in alto e K-means clustering non supervisionato in basso

Il cluster numero 2 è invece il cluster più eterogeneo per quanto riguarda i ruoli dei calciatori. Si trovano infatti per il 56% istanze etichettate come difensori, per il 9% attaccanti e per il 35% centrocampisti. E' ragionevole pensare che gli attaccanti presenti in questi cluster siano giocatori con pochi gol segnati e poche occasioni da gioco create durante la stagione 2023-2024. Osservando nomi e statistiche si nota, per esempio, i calciatori del Monza Milan Duric e Lorenzo Colombo, noti per non essere grandi goleador, ma per fare gran gioco di squadra attraverso contrasti e duelli aerei. Questo è anche confermato dalle statistiche a loro associate nel dataset originale. Il terzo cluster è formato dal 61% da difensori, dal 38% da centrocampisti e dal 1% da attaccanti. Come evidente e pronosticato in precedenza i centrocampisti sono sparsi in ugual modo nei tre cluster eterogenei, ossia che possiedono, anche se in bassa percentuale, le tre categorie più numerose. Ciò non accade nel quarto cluster, composto esclusivamente da Portieri. Questo è spiegato semplicemente dal fatto che le feature riferite ai portieri sono talmente tanto distanti e differenti rispetto a quelle degli altri giocatori che vengono uniti tutti all'interno di un unico cluster.

La qualità dei cluster formati attraverso questo algoritmo è stata valutata attraverso la silhouette media, un criterio di validazione interno che mostra quali istanze siano state ben clusterizzate e quali, in caso di silhouette negativa, siano state clusterizzate in maniera errata. La silhouette media riscontrata è pari a 0.25 e questo indica che i pattern sono stati assegnati ai loro cluster in modo ragionevolmente buono, ma ci sono alcune considerazioni da tenere presente: alcune istanze si trovano in punti di confine tra un cluster e l'altro ed hanno valori di silhouette veramente bassi, soprattutto per i cluster 2 e 3. Questo difetto è anche ben visibile in figura 9 dove alcuni punti blu risiedono nel cerchio rosso e altri in quello verde. Questo potrebbe essere un indicatore di incertezza, nonostante la rappresentazione consideri soltanto le prime due componenti principali.

4.2 Gaussian Mixture Model

A differenza dell'algoritmo K-means, che offre un approccio crispy al clustering, i cluster ottenuti attraverso modelli mistura offrono un approccio più flessibile e probabilistico per affrontare complessità nei dati che non si adattano a strutture di cluster chiaramente delineate. I modelli mistura per il clus-

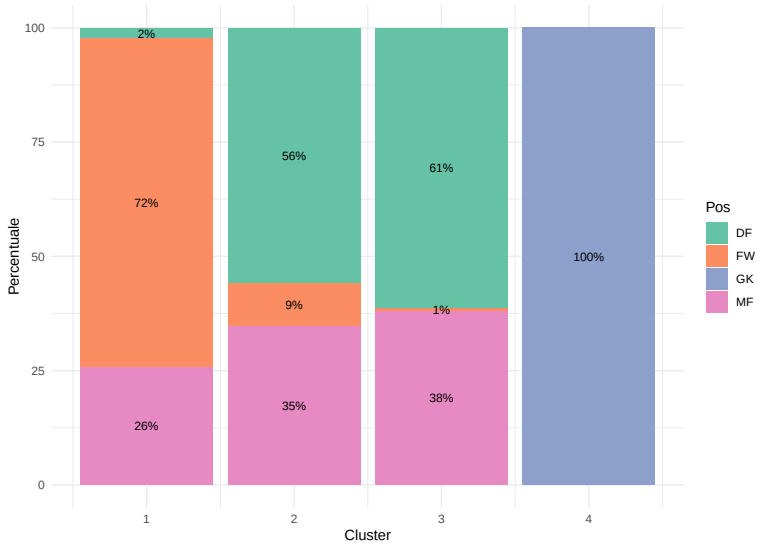


Figure 11: Percentuali di Pos in ciascun cluster(K Means)

tering sono una estensione dell'algoritmo EM che, attraverso la scomposizione della matrice di Varianza dei singoli cluster, permette di modificare la loro composizione attraverso un algoritmo iterativo a due stadi. In particolare permette di modificare la loro forma, il loro volume e il loro orientamento. I modelli mistura considerano ogni cluster come una distribuzione di probabilità, consentendo la rappresentazione di cluster con forme e dimensioni diverse, oltre a gestire in modo più robusto la presenza di dati con rumore o dati provenienti da distribuzioni multiple. In questo caso si ipotizza che la distribuzione congiunta di tutte le variabili considerate nell'analisi, ossia le prime undici componenti principali, sia multimodale e caratterizzata da un vettore delle medie, una media e una matrice di covarianza diversa per ogni cluster (casi più semplici assumono matrice di covarianza comune). Tramite l'algoritmo E-M vengono stimati diversi modelli mistura, ognuno con caratteristiche variabili date dalle diverse ipotesi sulla scomposizione di matrice di covarianza. Attraverso il criterio ICL (Integrated Completed Likelihood) è stato scelto il modello mistura avente forma, orientamente e volume differente per ogni cluster (modello VVV). Il numero di cluster ottimale è risultato essere pari a cinque, come evidenziato in figura 12.

Il modello comunica che le feature sono distribuite come una mistura di cinque Gaussiane con valore atteso e matrice di varianza e covarianza diversa per ogni componente. In particolare il modello VVV afferma che i cluster avranno diversa forma, diverso orientamento e diverso volume in R^{11}. Queste caratteristiche sono ben visibili nella proiezione a due dimensione espressa in figura 13.

Oltre alle diverse matrici di varianza e covarianza, l'algoritmo E-M stima anche i valori attesi delle diverse variabili per ogni sua componente del mistura. Grazie a ciò è possibile calcolare una misura, chiamata divergenza di Kullback-Leibler per verificare quanto i cluster ottenuti siano vicini tra loro. In tabella 10 sono infatti rappresentati i valori della divergenza di Kullback-Leibler.

Cluster	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	0				
Cluster 2	53.66581	0			
Cluster 3	42.85136	14.79639	0		
Cluster 4	3250.822	6212.315	7941.022	0	
Cluster 5	8.08826	21.40965	21.16639	2276.711	0

Table 10: Divergenze di Kullback-Leibler

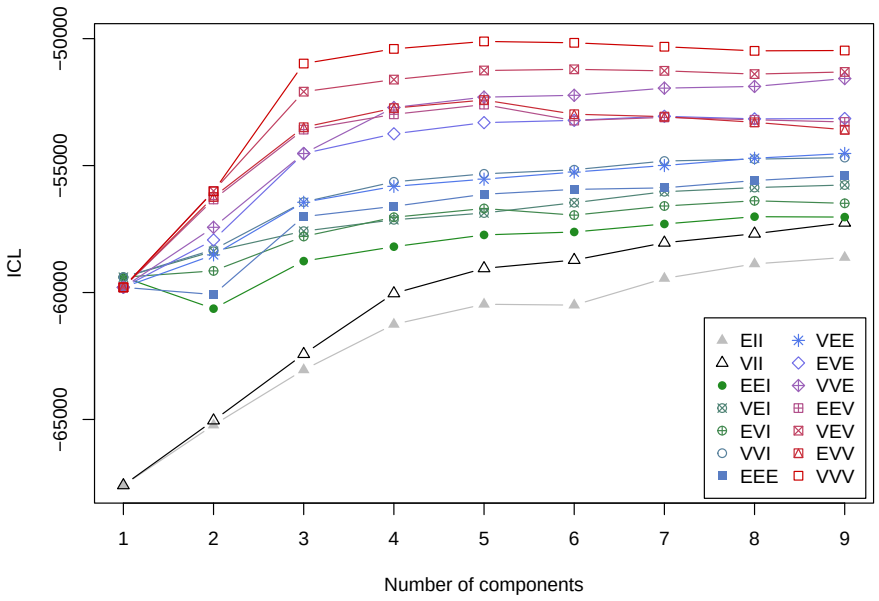


Figure 12: Scelta del modello e del numero di cluster secondo ICL

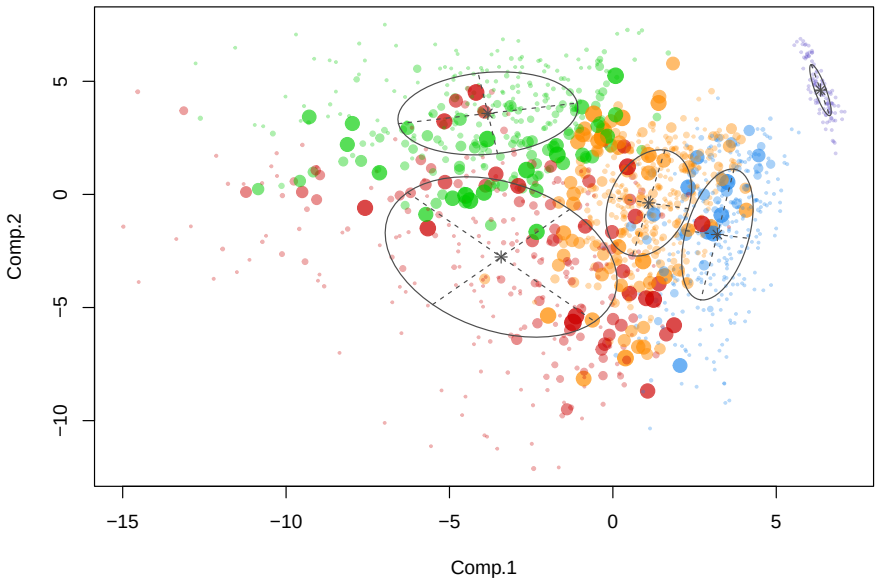


Figure 13: Rappresentazione bidimensionale delle forme e dell'orientamento dei cinque cluster ottenuti attraverso il modello mistura

Come osservabile tutti i cluster sono ben distanti dal cluster 4. Si può notare come il cluster 1 e il cluster 5 siano molto vicini tra loro. Come già svolto nella sezione 4.1, si osservano le percentuali di istanze per posizione all'interno di ciascun cluster. Il cluster 4, quello più lontano da tutti, anche in questo caso è composto interamente da 110 portieri. Il quinto cluster, tuttavia, è composto in egual modo da difensori e centrocampisti, con una piccola percentuale di attaccanti. Osservando le istanze raggruppate è possibile notare come i centrocampisti presenti siano tutti centrocampisti difensivi, le cui statistiche risaltano prevalentemente azioni di difesa a discapito di gol, tiri in porta e altri attributi offensivi. Il cluster 4 è composto in maggior parte da attaccanti e per il 19% da centrocampisti, di cui la maggioranza facente parte del sotto-ruolo del trequartista. Per quanto riguarda il cluster 1 si nota una prevalenza di difensori (98%). L'algoritmo è in grado di raggruppare ed individuare tutti quei giocatori con caratteristiche fortemente difensive e distinguerli dai centrocampisti, cosa che non avveniva con il K-Means. Ciò nonostante tutti i difensori con valori intermedi tra difesa e attacco vengono raggruppati all'interno del cluster 5. Il cluster 3 è formato prevalente da attaccanti (79%) e da centrocampisti con qualità offensive. Osservando i dati si nota che a far parte di questo cluster ci sono giocatori come Kai Havertz e Thomas Müller, ossia centrocampisti che giocano prevalentemente nella trequarti offensiva, concludendo la stagione con un numero piuttosto elevato, per essere un centrocampista, di gol e assist. Per concludere, analizzando il cluster 2 esso è composto prevalentemente da centrocampisti (60%). I risultati appena evidenziati sono rappresentati in figura 14.

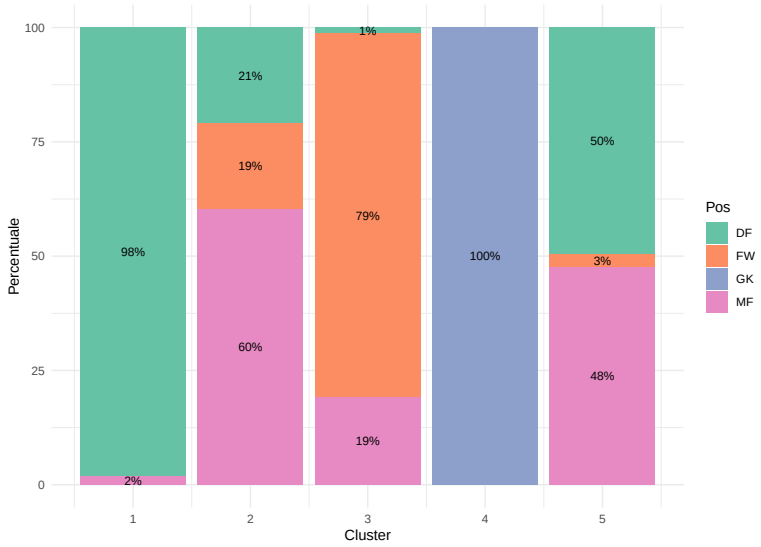


Figure 14: Percentuali di Posizioni in ogni cluster (Mixture Models)

In definitiva è possibile utilizzare la stessa metrica vista in sezione 4.1 per valutare la bontà dei cluster, ossia la silhouette media. I cluster 4 e 5 hanno un valore di Silhouette media piuttosto elevato, rispettivamente pari a 0.58 e 0.27. Tuttavia negli altri cluster è presente molta incertezza, tanto da ottenere una Silhouette media del clustering pari a 0.17, non ottimale; rispecchia perfettamente l'incertezza dei gruppi formati.

In sintesi l' algoritmo **E-M** è un'algoritmo più flessibile rispetto al **K-Means**, il quale forza i cluster ad assumere una forma sferica (ipersferica nel caso in esame). Essendo più flessibile l'algoritmo E-M è in grado di modellare forme diverse di cluster, tuttavia questa flessibilità può portare a gruppi meno stabili. Inoltre questo è molto influenzato dal rumore e dagli outlier poichè tenta di modellare anche le aree a bassa densità di dati. Un esempio sono i centrocampisti, con alcune statistiche simili alla media dei pari ruolo, mentre altre da attaccante puro o difensore insuperabile. Questo è ben visibile in figura 13 dove l'ellisse più grande, a sinistra, copre un'area del piano densamente scarsa di punti. Un ulteriore dubbio provocato da valori di silhouette così bassi ricadono sulla reale distribuzione congiunta dei dati,

che con molta probabilità, non è gaussiana. Nonostante ciò entrambe le clusterizzazioni effettuate con i due differenti algoritmi portano a risultati facilmente interpretabili.

5 Conclusioni e Ulteriori Sviluppi possibili

Gli algoritmi di regressione hanno portato a risultati ottimali per giocatori dal valore medio, mentre hanno di certo sottostimato valori di giocatori importanti. Un esempio può essere Vinicius Junior che ha un valore di mercato reale di 180 milioni di euro, pari a quelli che possiamo considerare outlier come Jude Bellingham o Erling Haaland. Con i metodi utilizzati, invece gli è stato stimato un valore di mercato pari a 51 milioni. La ragione di questa sottostima, come detto nella sezione 2.1.4 può derivare dal fatto che il valore di mercato di un giocatore è determinato da fattori anche extra campo. Le tecniche di classificazione hanno portato ad un' accuracy del 90%. Un risultato notevole se comparato a quelli ottenuti con algoritmi di clustering. Quest'ultimi infatti sono riusciti solo parzialmente ad organizzare i dati in strutture robuste in relazione alla posizione dei giocatori, mostrando un alto grado di incertezza. Le ragioni di questa differenza possono risiedere nelle ipotesi sottostanti agli algortimi utilizzati. La necessità che i dati siano ben separati tra loro e che gli algoritmi utilizzati cerchino di organizzare i dati in strutture circolari o ellittiche penalizza il task. Inoltre la presenza di notevoli outlier può portare ancora maggiore incertezza nella formazione dei cluster. Al contrario viene premiata una maggiore flessibilità da parte degli algoritmi di classificazione che attraverso linee di decisione formate da iperpiani si adattano meglio al contesto studiato perchè più robusti nella gestione degli outlier. Un'ulteriore considerazione sugli algoritmi di clustering è riferita al fatto che quest'ultimi appartengono all'*Unsupervised Learning*. Come tali si muovono nel dataset per cercare pattern che non per forza devono corrispondere con le etichette corrispondenti ai valori possibili della variabile *"Pos"*. Ulteriori sviluppi dello studio possono comportare l'utilizzo di reti neurali nel task di classificazione oppure ad approfondire il legame tra gli algoritmi di clusterizzazione e i pattern individuati nei dati. Altri sviluppi possibili potrebbero essere quello di utilizzare gli stessi algortimi studiati per dati provenienti da altri campionati, ad esempio il nascente campionato saudita, oppure concentrarsi solo su quei calciatori la cui posizione è missclassificata ai fini dell'individuazione di possibili talenti.

Riferimenti bibliografici

[1] Bernd Bischl et al. *mlrMBO: A Modular Framework for Model-Based Optimization of Expensive Black-Box Functions*. 2017. URL: https://arxiv.org/abs/1703.03373.

[2] FBref. *FBref Sports Stats*. 2024. URL: https://fbref.com/it/.

[3] Plotly Technologies Inc. *Collaborative data science*. 2015. URL: https://plot.ly.

[4] Kamil Slowikowski. *ggrepel: Automatically Position Non-Overlapping Text Labels with 'ggplot2'*. https://ggrepel.slowkow.com/, https://github.com/slowkow/ggrepel. 2024.

[5] Transfermarket. *Trasnsfermarket Market Value Player*. 2024. URL: https://www.transfermarkt.com/.

[6] Jason Zivkovic. *worldfootballR: Extract and Clean World Football (Soccer) Data*. R package version 0.6.5.0004. 2024. URL: https://github.com/JaseZiv/worldfootballR.