Renzetti Franesco 807449

Machine Learning M (Università degli Studi di Milano-Bicocca)

Tecniche di Machine Learning: analisi e previsioni dei risultati delle partite di Tennis

Francesco Renzetti

Keywords: K-means, K-medoids, DBSCAN, Dendrogramma, Random Forest, Support Vector Machine, Reti Neurali

23 Giugno 2022

Il tennis è uno sport molto popolare. Da tempo i ricercatori hanno cercato di prevedere l'esito delle partite di tennis utilizzando dati forniti dalle statistiche relative a incontri effettuati in passato. In questo articolo ho utilizzato algoritmi non supervisionati, quali K-means, K-medoids e DBSCAN ottenendo un raggruppamento significativo dei risultati delle partite riguardanti l'Australian Open del 2013. Successivamente ho utilizzato algoritmi di classificazione supervisionati, quali la Random Forest, la Support Vector Machine e le Reti Neurali allo scopo di prevedere i risultati dello US Open del 2013 partendo dai risultati dell'Australian Open. Tutte le misure di performance confermano una maggiore efficienza della Random Forest, in quanto ho ottenuto un'accuracy del 93%. Infine ho preso un altro dataset e l'ho ristrutturato in modo tale da ottenere tutte le statistiche possibili dei giocatori in tutti i tornei giocati sul cemento nel 2013. Con i metodi di clustering K-means e col dendrogramma, sono riuscito a raggruppare i migliori giocatori su quella superficie nell'anno 2013 che, in alcuni casi, si sono rivelati essere i tennisti che hanno raggiunto le fasi finali dell'Australian Open e dello US Open.

1 Introduzione

Il tennis è uno dei giochi popolari giocati e guardati in tutto il mondo. È uno sport praticato individualmente (singolo) o in squadra di due (doppio). Ci sono quattro principali competizioni di tennis del Grande Slam che si svolgono ogni anno: Australian Open, French Open, Wimbledon e US Open. Questi quattro tornei di tennis del Grande Slam sono i più famosi in tutto il mondo. Inutile dire che anche le superfici dei campi di questi affascinanti eventi di tennis sono diverse. Australian e US Open si giocano su campi in cemento, French Open su terra battuta e Wimbledon su erba. Ogni superficie del campo ha le sue caratteristiche e crea variazioni nel rimbalzo e nella velocità della palla. I campi in terra battuta producono un ritmo della palla più dolce e un rimbalzo altrettanto preciso con effetti extra. I campi in cemento producono una palla più veloce e rimbalzi molto precisi. I campi in erba favoriscono movimenti della palla più rapidi con l'aggiunta di rimbalzi imprevedibili. In questo articolo, per effettuare le analisi, ho preso in considerazione solo le partite in singolare, giocate da atleti maschi e che si svolgono sul cemento. In tal modo ho potuto analizzare partite svolte in condizioni simili.

I dataset di riferimento che ho utilizzato per l'analisi sono fruibili ai seguenti link: https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics (AusOpen-men-2013.csv, USOpen-men-2013.csv) e https://www.kaggle.com/gmadevs/atp-matches-dataset (atp-matches-2013.csv). I primi due dataset sono composti da 127 istanze e 42 features; ogni istanza si riferisce ad una partita del rispettivo torneo. "AusOpen-men-2013.csv" è quello che ho utilizzato inizialmente per effettuare la cluster analysis. Ho utilizzato poi questo dataset come training set per andare a prevedere i risultati dello US Open sul test

set "USOpen-men-2013.csv".

Il dataset atp-matches-2013.csv, composto da 2959 istanze e 49 features, contiene le informazioni sulla maggior parte delle partite svolte nel 2013 in molti dei tornei, sia principali che meno importanti, ed è stato da me utilizzato per effettuare la Cluster analysis finale.

1.1 Obiettivi

- La Cluster analysis è un metodo statistico per processare i dati attraverso il raggruppamento degli elementi di un insieme, a seconda delle loro caratteristiche, in classi non assegnate a priori che siano il più eterogenee possibili. In questo articolo l'ho utilizzata per due scopi: il primo è andare a verificare come le statistiche delle partite siano dei buoni predittori dei risultati degli incontri, andando a confrontare i gruppi ottenuti dalla Cluster analysis con il risultato delle partite.
  Il secondo scopo è quello di indagare il fatto che le superfici di gioco sono fondamentali per il rendimento dei giocatori. Raggruppando infatti i tennisti che hanno buone statistiche sul cemento, è possibile osservare che alcuni di essi, nonostante non abbiamo un ranking generale elevatissimo, nei tornei giocati su questa superficie hanno avuto un ottimo rendimento durante l'anno.
- La fase di classificazione mira invece a trovare il miglior metodo/modello capace di prevedere i risultati delle partite dello US Open sulla base dei risultati ottenuti nelle partite dell'Australian Open. In questo articolo ho utilizzato tre metodi di classificazione, che andrò ad analizzare nello specifico più avanti, per poi selezionare quello/i aventi un'accuratezza maggiore.

2 Preprocessing

2.1 Analisi delle variabili

- **"AusOpen-men-2013.csv" e "USOpen-men-2013.csv":**
  Come già anticipato nella sezione 1, questi due dataset contengono tutte le statistiche riguardanti i risultati delle partite rispettivamente dell'Australian e dello US Open dell'anno 2013. Entrambi contengono lo stesso numero di istanze (127 partite giocate durante il torneo) e le stesse features (42 statistiche riguardanti ogni singola partita del giocatore 1 contro il giocatore 2). Questo significa che ogni feature che sono andato a togliere nel dataset dell'Australian Open che ho utilizzato come train, ho dovuto eliminarla anche dal dataset dello US Open usato come test. Il preprocessing quindi è speculare per i due dataset. Inizialmente ho tolto dall'analisi tutte le variabili e le osservazioni contenenti troppi valori mancanti poiché non contengono alcun tipo di informazione utile. Eliminando però una variabile riguardante il giocatore 1, ho dovuto toglierla anche per il giocatore 2. Dopodiché ho standardizzato le variabili per renderle più facilmente confrontabili. Successivamente, al fine di rendere le variabili più facilmente leggibili e più trattabili, ho unito le features in comune ai due giocatori sottraendo l'osservazione riferita al giocatore 1 a quella del giocatore 2. Ottenendo come risultato un valore positivo, significa che il giocatore 1 ha ottenuto un punteggio maggiore della relativa variabile rispetto al giocatore 2. In questo modo ho ridotto il dataset da 42 variabili totali alle seguenti 19:

Player1	Nome del giocatore 1
Player2	Nome del giocatore 2
Round	Turno della partita
Results	1 se il giocatore 1 vince
FNL	nº di game finali vinti
FSP	% di prime al servizio
FSW	Prime al servizio vincenti
SSP	% di seconde al servizio
SSW	Seconde al servizio vincenti
ACE	Ace vincenti
DBF	Doppi falli commessi
BPC	Break points creati
BPW	Break points vinti
NPA	nº di net
NPW	Punti vinti col net
TPW	Punti vinti totali
ST1-3	Risultati dal set 1 al 3

Successivamente ho tolto dall'analisi le variabili "Player1", "Player2" e "Round" in quanto superflue per il lavoro.
Osservando la correlazione presente tra le variabili esplicative (escludendo quindi "Result"), ho osservato, tramite il metodo di Spearman, la multicollinearità tra le variabili, in particolare è presente una correlazione negativa perfetta tra le variabili "FSP" e "SSP"; ho deciso quindi di togliere "FSP" dall'analisi. La figura 1 fornisce un'indicazione sulla collinearità presente tra le variabili.

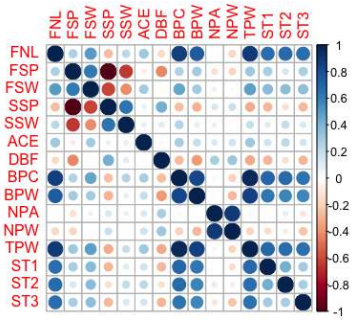

Figure 1: *Multicollinearità presente tra le variabili in esame.*

Infine ho eliminato anche la variabile "FNL" perché indica il numero di game vinti dai giocatori e, quasi sempre, il tennista che ne conquista di più vince anche la partita. Ciò porterebbe a delle classificazioni fin troppo precise.
Nel resto dell'analisi ho deciso di considerare tutte le altre variabili.

- **"atp-matches-2013.csv":**
  Questo dataset contiene la maggior parte delle partite svolte nell'anno 2013 nei tornei ATP. Ho filtrato il dataset considerando solo le partite svolte sul cemento (1700 istanze). Le 49 variabili rappresentano tutte le statistiche degli incontri di ogni torneo (39 tornei diversi).
  Dopo aver eliminato le variabili con troppi valori mancanti, ho ristrutturato completamente questo dataset calcolando solo le percentuali utili raggruppate per ogni giocatore per andare poi a trovare, tramite Cluster analysis, i migliori giocatori su cemento che hanno avuto, durante il corso dell'anno, un ranking minore di cinquanta. Considero solo questi tennisti per una semplificazione grafica e perché un giocatore con ranking troppo elevato difficilmente riesce ad arrivare nelle fasi finali dei tornei del Grande Slam.
  Le variabili che ho ottenuto sono le seguenti riassunte nella tabella sottostante:

player-name	Nome del giocatore
1stServeWon	% di prime al servizio vincenti
player-ace-p	% di ace
player-df-p	% di doppi falli
bpSaved	% di break point salvati

3 Australian Open Clustering

3.1 Iterative Distance-Based

Il primo approccio è basato sull'utilizzo degli algoritmi K-means e K-medoids. I due algoritmi sono simili, infatti ho utilizzato per entrambi la distanza Euclidea; essi cercano di minimizzare l'errore quadratico medio, ovvero la distanza tra i punti di un cluster e il punto designato per esserne il centro. Nel K-means il punto centrale (centroide) è rappresentato dal baricentro (posizione media) di tutti i punti nel cluster. Nel K-medoids è utilizzata la mediana, infatti il punto centrale (medoide) è rappresentato da uno dei dati osservati.

Utilizzando la regola del gomito, ho notato che il numero ottimale di cluster per entrambi i metodi è due.

Dai risultati è possibile notare come gli algoritmi hanno cercato di raggruppare nel cluster 1 le partite vinte dal giocatore 2, mentre nel cluster 2 le partite vinte dal giocatore 1. Il K-means ha inoltre classificato 42 istanze nel cluster 1 e 56 nel cluster 2. Il K-medoids ne ha raggruppate invece 49 in entrambi i cluster.

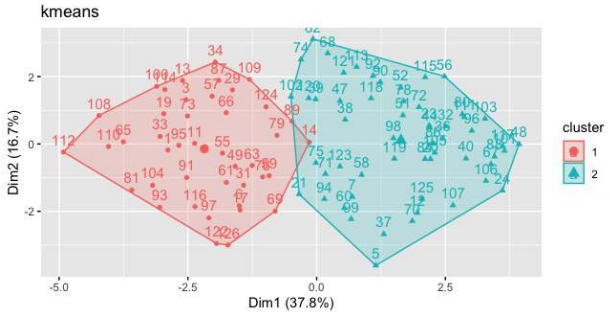

Figure 2: *Grafico relativo ai cluster individuati dal K-means.*

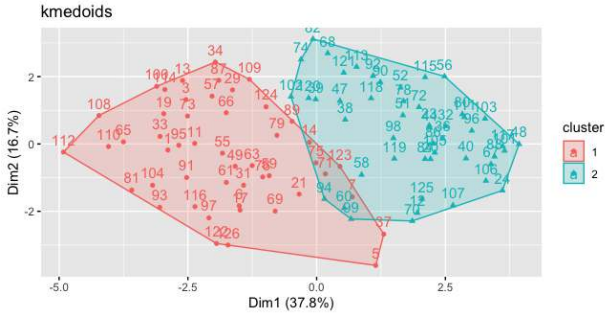

Figure 3: *Grafico relativo ai cluster individuati dal K-medoids.*

Ho confrontato i risultati forniti dai due algoritmi sulla base di due metodi: il primo è un criterio interno, ovvero la Silhouette che è un indice in grado di valutare quali punti sono stati assegnati in modo corretto ai cluster e quali no. Il valore medio della Silhouette ottenuto per il K-means è di 0.26, mentre per il K-medoids è di 0.24. Entrambi i valori sono maggiori di zero e quindi gli algoritmi hanno creato una partizione attendibile anche se il K-means, nel problema in esame, sembra avere una maggiore precisione. La Silhouette infatti considera il fatto che l'algoritmo K-means ha ottenuto un'area di intersezione tra i due cluster più piccola.

Questo risultato non è però confermato dal secondo metodo utilizzato, che è un criterio esterno cioè la matrice di confusione. Conoscendo i veri cluster è possibile costruire questa matrice che restituisce una rappresentazione dell'accuratezza della classificazione. Per quanto riguarda il K-means si è ottenuta un'accuratezza del 90%, mentre per il K-medoids del 95%. Questo risultato dimostra che è preferibile il K-medoids in quanto è più robusto al rumore e agli outliers rispetto al K-means.

3.2 DBSCAN

Questo algoritmo è basato sul concetto di densità delle istanze nello spazio. L'idea su cui si basa è quella di andare a cercare, per ciascun punto, un'area circostante

(neighborhood) che deve contenere un minimo di punti (spesso indicati con MinPts), dato uno specifico raggio (spesso indicato con Eps). I punti aventi densità superiore ad un certo numero di MinPts vengono chiamati **core point**. Se la densità attorno ad un punto non supera il numero MinPts richiesto e se un punto core si trova ad una distanza minore di Eps, allora quei punti vengono chiamati **border point**. I punti che non rientrano in queste classificazioni vengono considerati come rumore e si chiamano **noise point**. Siccome questo algoritmo è fortemente sensibile alla scelta dei parametri iniziali, ho utilizzato l'Automated Machine Learning per trovare i valori ottimali di Eps e MinPts. Prima di tutto ho usato la regola del gomito per osservare un range di valori ottimali per Eps, ovvero $3.5-4.5$ e, per ragioni computazionali, ho considerato minPts in un range di $10-20$. Ho poi ripetuto l'algoritmo nei due range considerati e sono andato a vedere quale combinazione di valori di Eps e MinPts va a massimizzare la media della Silhouette. In seguito al fatto che diverse combinazioni raggiungono questo risultato, ho considerato solo la prima di queste. I valori utilizzati sono riassunti nella tabella seguente:

Eps	4
MinPts	12
Silhouette media	0.19448

Il risultato finale ottenuto è osservabile nella figura 4:

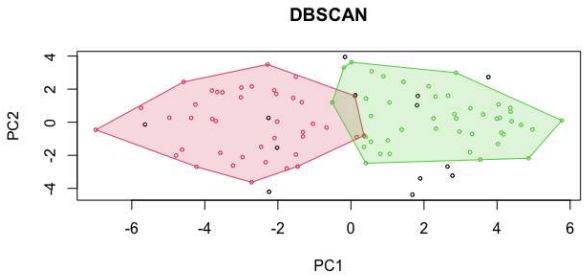

Figure 4: *Grafico relativo ai cluster individuati dal DBSCAN.*

I cluster trovati dall'algoritmo DBSCAN sembrano essere simili a quelli trovati dal K-means: entrambi classificano in base ai risultati delle partite, ma la differenza sostanziale sta nel fatto che il DBSCAN esclude dall'analisi tutti i punti che considera come outliers. L'algoritmo ha infatti inserito 41 punti nel cluster 1, 44 nel cluster 2 e 13 sono stati considerati outliers.

4 Best Player Clustering

In questa sezione sono andato a ristrutturare completamente il dataset " atp-matches-2013.csv" tenendo in considerazione solo le variabili che ho calcolato, visibili nella sezione 2, al fine di ottenere dei cluster contenenti i migliori giocatori di Tennis su cemento nel 2013. A questo scopo ho utilizzato due metodi: K-means e il Dendrogramma.

4.1 K-means

Inizialmente, tramite il metodo del gomito, ho osservato che il numero di cluster ottimo è pari a 4. Successivamente ho applicato l'algoritmo k-means al mio dataset ottenendo il seguente risultato:

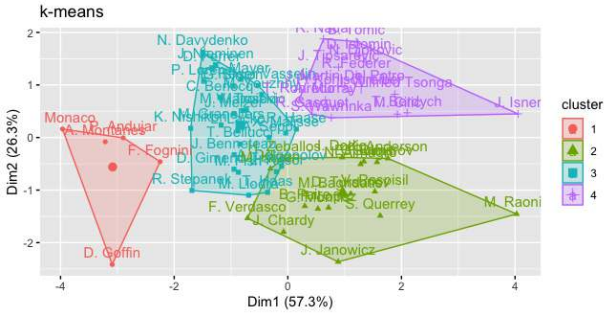

Figure 5: *Grafico relativo ai cluster individuati dal K-means.*

Nel cluster viola sono raggruppati i migliori giocatori, in quello azzurro quelli che hanno fornito prestazioni inferiori ai precedenti e, a scalare, nel cluster verde e rosso i tennisti sempre meno performanti.

4.2 Dendrogramma

Il Dendrogramma fa parte degli algoritmi di clustering gerarchici di tipo agglomerativo: ogni elemento appartiene ad un cluster distinto nel primo livello. Tutti i gruppi sono fusi a coppie in base alla vicinanza nei livelli successivi fino a quando non si ottiene un'unica configurazione gerarchica che racchiuda tutti i cluster precedenti. Diversamente dal K-means quindi, il Dendrogramma tiene traccia di come i punti sono stati uniti o separati dai gruppi. Anche in questo caso ho utilizzato la distanza euclidea per indagare le similarità presenti nei dati.

Il risultato ottenuto è possibile vederlo nella seguente figura:

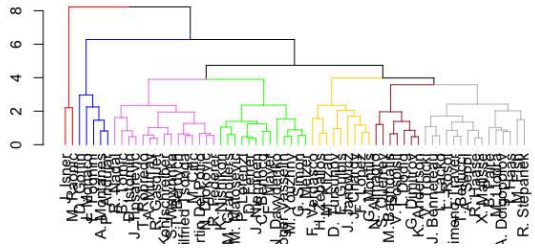

Figure 6: *Grafico relativo ai cluster individuati dal Dendrogramma.*

Si può notare come i cluster identificati siano ben riconoscibili, il che implica maggior similarità all'interno dei gruppi trovati e diversità tra di essi.

4.3 Interpretazione dei risultati

È interessante notare come il Dendrogramma ha trovato un numero maggiore di cluster, questo perché il K-means è sensibile alla scelta del parametro iniziale.

Dai cluster risultanti è possibile notare che i giocatori aventi un rendimento migliore su cemento nell'anno 2013 sono stati messi all'intero dello stesso gruppo (quello viola per entrambi) eccetto per un giocatore (Isner) che il Dendrogramma ha inserito in un altro cluster.

È interessante osservare inoltre che alcuni tennisti aventi un ranking generale relativamente basso, il quale tiene in considerazione tutte le altre partite giocate su superfici diverse, sono stati inseriti da entrambi gli algoritmi nel cluster viola contenente i migliori giocatori. Questo significa che nel 2013 questi giocatori hanno avuto ottime statistiche sul cemento che li hanno portati ad avere buoni risultati nei tornei di quell'anno, come è possibile vedere dalle tabelle che mostrano sia il ranking di questi giocatori sia i tornei in cui sono arrivati almeno ai quarti di finale.

	B. Tomic (40)
Sidney	Winner
Marsille	Quater finals

	D. Istomin (42)
Memphis	Quarter finals
Brisbane	Quater finals
Atlanta	Quater finals
St. Petersburg	Semi finals

	T. Robredo (18)
Winston-Salem	Quarter finals
US Open	Quater finals

	P. Kohlschreiber (19)
Valencia	Quarter finals
Bejing	Quater finals
Auckland	Final

Inutile dire che giocatori del calibro di Djokovic e Nadal, vincitori rispettivamente dell'Australian Open e dello US Open nel 2013, sono stati classificati sempre in questo gruppo (viola).

5 Classificazione

In questa sezione ho deciso di utilizzare il dataset dell'Australian Open come training set e quello dello US Open come test set. I metodi utilizzati per andare a prevedere i risultati dello US Open sono: Random Forest, Support Vector Machine e Reti Neurali. Tutti questi metodi sono stati addestrati nel training set tramite repeated 10-fold cross-validation in modo tale da utilizzare gli iperparamentri in grado di minimizzare il generalized error.

5.1 Random Forest

Il primo metodo di classificazione implementato è la Random Forest. Con questo algoritmo si costruisce una moltitudine di alberi decisionali durante l'addestramento. In particolare, ho utilizzato come valori in input $ntree = 62$ e $mtry = 3$. Come è possibile vedere dalla figura, l'albero ha fornito automaticamente un peso alle variabili in base all'importanza delle stesse nel prevedere i risultati delle partite:

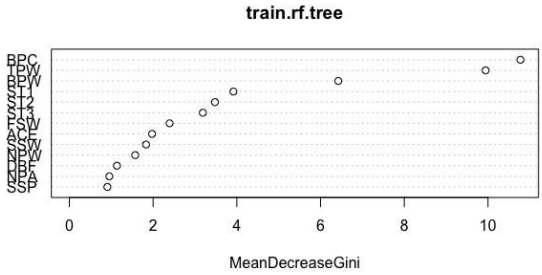

Figure 7: *Il grafico evidenzia l'importanza delle variabili attribuite dalla Random Forest.*

Le variabili maggiormente considerate in ordine di importanza sono: BPC, TPW e BPW.

In questo modo ho ottenuto ottimi risultati in termini di robustezza, come è possibile osservare dalla tabella:

	Value
Generalized-Error	0.092
Empirical-Error	0.102
Test-Error	0.07
Accuracy	93%

L'errore di generalizzione e l'errore empirico sono molto simili, per questo motivo non si incombe in problematiche di overfitting o underfitting e l'algoritmo generalizza molto bene.

5.2 Support Vector Machine

Il secondo algoritmo utilizzato è la Support Vector Machine. L'idea di base dell'algoritmo è individuare un iperpiano ottimale che separa tra loro dati che sono

linearmente separabili. È possibile estendere questo concetto ai casi in cui i dati non sono linearmente separabili tramite delle trasformazioni dei dati in un nuovo spazio attraverso le funzioni di Kernel.

Prima di applicare l'algoritmo, ho deciso di effettuare una selezione delle variabili tramite Lasso al fine di ottenere una migliore accuratezza nelle previsioni. La Support Vector Machine infatti non tiene conto, come invece fa la Random Forest, dell'importanza delle variabili e non è in grado di gestire la multicollinearità delle stesse. Un primo step di selezione delle features può aiutare a risolvere questi probelmi. Ho preso quindi in considerazione solo SSW, BPC e TPW.

Nel caso in esame, inoltre, i dati non sono linearmente separabili e quindi ho dovuto ricorrere al "Kernel trick". Ho deciso di utilizzare prima un kernel polinomiale e poi un kernel radiale arrivando a scegliere di applicare quest'ultimo in quanto fornisce una robustezza maggiore. Ho scelto i parametri di input ottimi, sempre tramite repeated 10-fold cross-validation, ottenendo $cost = 0.25$ e $sigma = 0.739$. Il primo parametro è la penalità e ci indica quanto errore massimo può essere ammesso. Il secondo definisce quanta influenza ci deve essere tra i punti e la linea di separazione; al crescere del valore di sigma, i punti vicini avranno un' influenza elevata, quando quest'ultimo è basso, anche i punti lontani contribuiranno a ottenere il confine di decisione.

Gli errori ottenuti nelle varie fasi sono riportati nella tabella:

	Value
Generalized-Error	0.06
Empirical-Error	0.06
Test-Error	0.08
Accuracy	92%

L'errore di generalizzazione e l'errore empirico sono approssimativamente uguali, questo dimostra l'ottima robustezza dell'algoritmo ottenuta. Si è verificata inoltre un'accuratezza leggermente inferiore rispetto alla Random Forest, ma comunque molto alta.

5.3 Reti Neurali

Le Reti Neurali sono uno dei metodi più importanti del machine learning il cui nome e struttura sono ispirati al cervello umano. Sono composte da livelli di nodi che contengono un livello di input, uno o più livelli nascosti e un livello di output. Ciascun nodo, o neurone artificiale, si connette ad un altro e ha un peso e una soglia associati. Se l'output di qualsiasi singolo nodo è al di sopra del valore di soglia specificato, tale nodo viene attivato, inviando i dati al successivo livello della rete. In caso contrario, non viene passato alcun dato al livello successivo della rete. La potenzialità di questo metodo sta nel creare una struttura computazionale densa di connessioni tra i neuroni, che utilizzi come elementi di attivazione funzioni non lineari.

In questo caso ho deciso di utilizzare come nella Random Forest tutte le variabili presenti nel dataset al fine di addestrare la rete su tutte le informazioni in possesso. Ho intolre utilizzato ancora la repeated 10-fold cross-validation al fine di trovare la dimensione ottima dello strato nascosto che massimizzi l'accuracy nel validation set. Di conseguenza ho costruito la rete neurale con un unico Hidden Layer formato da 3 neuroni con un tasso di decadimento (decay) di 0.6. Il risultato ottenuto si può vedere nella figura 8.

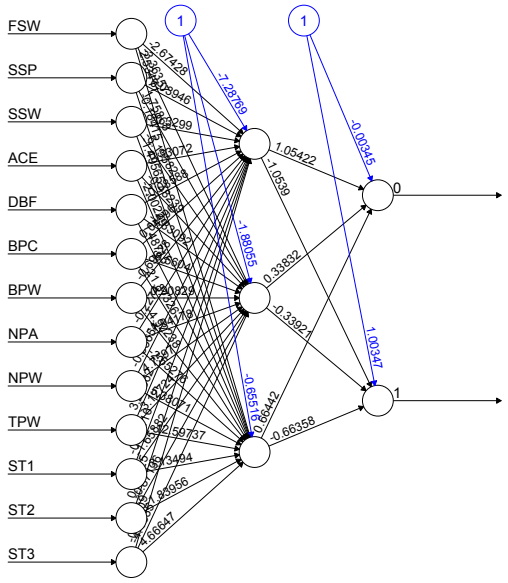

Figure 8: *Grafico della rete neurale.*

È interessante notare come i pesi iniziali dati alle variabili sono più elevati tanto più la variabile è ritenuta importante. L'ottimizzazione dei pesi avviene tramite back-propagation applicando l'algoritmo BFGS usato di default dalla funzione nnet.

Le variabili a cui sono stati assegnati pesi maggiori sono riassunte nella figura 9.

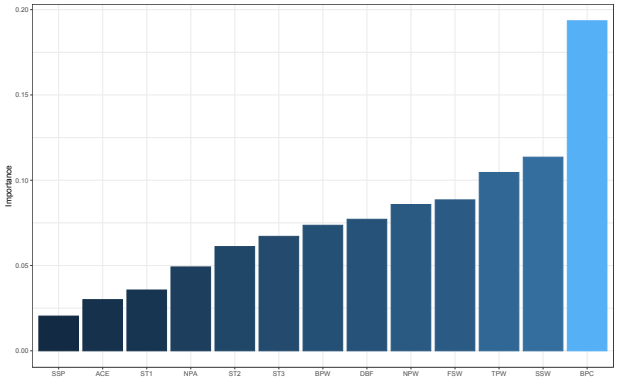

Figure 9: *Il grafico evidenzia l'importanza attribuita alle variabili dalla rete neurale con i relativi pesi.*

Coerentemente con la Random Forest, la variabile a cui è stato assegnato un peso maggiore è BPC. Nella

tabella sono infine riportate le stime degli errori:

	Value
Generalized-Error	0.03
Empirical-Error	0.06
Test-Error	0.11
Accuracy	89%

Ancora una volta sono confermate la robustezza dell'algoritmo e l'ottima accuratezza trovata.

5.4 Conclusioni

Gli algoritmi utilizzati nella prima fase di clustering hanno evidenziato un raggruppamento significativo dei risultati delle partite. Le matrici di confusione hanno sottolineato una migliore efficienza dell'algoritmo K-medoids (95% di accuratezza). Dai risultati ottenuti posso quindi concludere che le statistiche di gioco utilizzate durante l'analisi sono degli ottimi predittori dei risultati delle partite.

Nella seconda fase ho ottenuto un raggruppamento dei migliori giocatori su cemento coerente coi risultati e con le statistiche ottenute dagli stessi negli incontri giocati nel 2013.

I risultati ottenuti con la classificazione hanno evidenziato una precisione massima del 93%, tramite la Random Forest, nel prevedere i risultati delle partite dello US Open. In ogni caso tutti gli algoritmi utilizzati hanno fornito un'ottima robustezza e un'elevata accuratezza delle previsioni.

Una possibile estensione futura potrebbe essere la costruzione di modelli che tengano conto anche dei diversi terreni di gioco.

References

(N.d.[b]). URL: https://www.atptour.com/en/scores/results-archive.

Classification and Regression Training (2019). URL: https://cran.r-project.org/web/packages/caret/caret.pdf.

Feed-Forward Neural Networks and Multinomial Log-Linear Models (2022). URL: https://cran.r-project.org/web/packages/nnet/nnet.pdf.

M. Mohri A. Rostamizadeh, A. Talwalkar (2018). *Foundation of Machine Learning*. MIT Press.

T. Hastie R. Tibshirani, J.Friedman (2017). *The Elements of Statistical Learning*. Springer.

Tibshirani, R. (2013). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological), Vol. 58, No. 1 (1996), pp. 267-288*.