Web Intelligence 2020

Predizione del vincitore di Australian Open

Ivanoiu Alexandru Paul - 870383

Progetto :

Il progetto di Web Intelligence 2019/2020 prevede di predire il vincitore di Australian Open 2020 simulando il torneo creando i singoli incontri dal primo torneo alla fine, per poi restituire il nome del giocatore. I dataset usati per il progetto si possono trovare a questo link: http://tennis-data.co.uk/alldata.php.

Le due possibilità per il progetto erano : (Preso direttamente da moodle)

- Predict the winner of the 2020 Australian Open by simulating the draw match by match
- Design a betting strategy (how much to bet and on which matches) on the basis of a prediction model

La consegna del progetto è stata leggermente modificata in quanto al momento dello sviluppo del progetto non era possibile sapere i giocatori che avrebbero giocato. Visto questo problema ho creato un analisi e dei modelli per poter simulare e prevedere il vincitore di Australian Open 2019. Un altro piccolo problema incontrato è stato quello del numero di partite del 2019 in quanto c'è stata una partita in meno cioè non più 128, ma 127 partite per il ritiro di una persona.

Scraping:

Altri dati che ho aggiunto al dataset fornito sono stati raccolti tramite web scaping. Ho utilizzato l'estensione "DataMiner" che mi permette di estrarre i dati dalle tabelle e salvarle in un file excel. Il sito da cui ho fatto scraping è www.tennisabstract.com da esso ho estratto dati riguardo la bravura dei giocatori e riguardo la loro data di nascita, poi ho raccolto dati anche sulla bravura per ogni tipo di campo.

Data processing :

Prendendo in considerazione i dati forniti, ho deciso di unire i dataset dal 2019 al 2015 perché poi negli anni successivi c'erano dei giocatori che non non esistevano più nei tornei più attuali come ad esempio: il torneo del 2018/19.

Features :

Una volta uniti i dataset le features del dataset si potevano dividere in 2 categorie :

1. Features che riguardavano il torneo
2. Features che descrivono le quotazioni dei principali siti di scommesse.

Elenchiamo le features della prima categoria e specifichiamo quale features sono state droppate e perchè :

1. Features:
    a. ATP
    b. Location
    c. Tournament
    d. Date
    e. Series
    f. Court
    g. Surface
    h. Round
    i. Best of
    j. Winner
    k. Loser
    l. Wrank
    m. Lrank
    n. WPts
    o. LPts
    p. W1….W5
    q. Lsets
    r. Comments

Su tutte queste features alcune sono state droppate perchè o non influiscono sull'esito delle partite o perché prevedono il futuro e questa cosa non va bene.
Esempio di features droppate : ATP / Location / Tournament / Date / Winner / Loser / Wrank/Lrank/WPts/LPts/W1….W5/Lsets/Comments

Le features della seconda categoria come abbiamo detto riguardano il betting per cui sono state droppate tutte visto che noi dobbiamo predire il vincitore.

Dataset usati :

Ho iniziato il progetto leggendo 6 dataset, il primo (ATP_dataset_2019_2015) in cui si trovano tutte le partite e tutte le features necessarie per il training e il test, nel secondo file(Dataset_2) si trovano delle feature in più riguardanti il Winner o il Player1.

Nel terzo file(Dataset_3) si trovano delle feature in più riguardanti il Loser o Player2 e nel quarto dataset invece troviamo esattamente i player di Australian Open 2019 che sono stati estratti dal dataset iniziale (ATP_dataset_2019_2015), gli ultimi 2 dataset contengono le

date di nascita dei player, questa cosa mi serve per fare la differenza tra la data del torneo e la data di nascita di ogni player per poi dopo poter trovare l'età.

Missing values :

Dopo l'unione dei dataset descritti prima ci sono stati dei missing values che riguardavano la bravura dei player. I valori NaN ho deciso di riempirli con il valore 0 visto che le features riguardano quanto bene hanno giocato i giocatori durante la loro carriera.

A questo punto non sapendo come hanno giocato i giocatori con valori NaN non potevo riempire i campi con altri valori.

Features di tipo categoriale :

Nel nostro dataset esistono delle features categoriali come ad esempio la feature "**Series**" che indica l'importanza del torneo. Per questa features ho deciso di creare un ordine visto che se un torneo è più importante c'è più interesse da parte del player.

L'ordine giusto della feature "**Series**" è : ATP250 - ATP500 - Masters 1000 - Masters Cup - Grand Slams. Questa feature è stata mappata negli interi [0:4]. La stessa storia vale per le seguenti feature: "**Surface**" e "**Court**".

Target value :

Il problema da affrontare è un problema di classificazione di tipo binario perchè vogliamo che il nostro modello prenda in input una tupla formata da 2 player e predica 0 se il player1 vince o 1 se il player2 vince. Siccome il dataset che ho non è strutturato in questo modo tutte le tuple appartengono alla classe 0 però visto che il player1 è sempre il vincitore e il player2 è il perdente.

Il metodo usato da me per risolvere questo problema è stato di creare la colonna target e impostata tutta a zero. A questo punto ho creato una funzione swap_values che mi permette di scambiare i player tra di loro e di mettere il target value di quella riga a 1 visto che adesso il player2 ha vinto.

Applicazione dei modelli :

Ho scelto 2 algoritmi da confrontare: **k-Nearest Neighbors** e **RandomForest.** La prima cosa che ho fatto è stata quella di creare dei modelli banali per creare un limite inferiore per l'accuracy.

Dopo il lancio dei classificatori l'accuracy per il limite inferiore è stata :

Random Forest	k-Nearest Neighbors
≈ 0.70	≈ 0.68

Una volta fatto questo ho introdotto le nuove features :
'EloL','HardRawL','ClayRawL','GrassRawL','hEloL','cEloL','gEloL','AgeL','EloW','HardRawW','ClayRawW','GrassRawW','hEloW','cEloW','gEloW','AgeW','AgeDif'.

Le features che riguardano l'age sono state calcolate in base alla data di nascita e in base alla data quando è stata giocata la partita.

AgeDif è una feature che prende le data di nascita del player1 e player2 e fa la differenza tra esse per farci capire chi è più vecchio tra i due giocatori. L'idea era se sei più vecchio hai più esperienza.

Dopo l'introduzione delle nuove feature e il lancio dei 2 modelli possiamo vedere nella tabella sotto dei miglioramenti riguardanti l'accuracy dei 2 modelli.

Random Forest	k-Nearest Neighbors
≈ 0.83	≈ 0.80

Nonostante il modello Random Forest è molto più complesso del k-Nearest Neighbors ho ottenuto un aumento nel accuracy di solo 0.03%.

L'elaborazione del torneo:

Ho iniziato la parte della creazione del torneo prendendo solo i dati del torneo di Australian Open 2019 per poter predire il vincitore.

L'idea usata per la creazione del torneo è prendere i giocatori del 2019 in un file diverso contenente solo il nome dei giocatori accoppiati per partite.

Il secondo passo è prendere proprio le partite del 2019 con le feature di ogni partita, prendere il modello e lanciarlo per ogni riga. I risultati vengono introdotti in un vettore che poi mi permette di selezionare il vincitore di ogni partita e creare i nuovi dati che serviranno per il prossimo Round. Questo ragionamento si ripete per ogni singolo Round finché non rimangono solo 2 player che giocano la finale di cui estraggo il nome del vincitore.

Ho creato le seguenti funzioni per poter creare in seguito le partite che usano i dati delle partite precedenti.

"**print_predictions**": ho creato questa funzione per poter predire per ogni partita il vincitore, le predizioni le salvo in ordine in un vettore che mi servirà poi per estrarre i dati da altri dataset e creare dataset nuovi per la prossima fase del torneo.

“**Fun_retu**”: ho creato questa funzione per poter decidere il vincitore di ogni singola partita

“**Ret_data_players**”: ho creato questa funzione per poter ritornare i dati(le features) delle colonne dei player vincitori per poi poter salvare questi in un nuovo dataset che userò nella prossima partita.

“**net_dataset**”: ho creato questa funzione per poter salvare proprio il nome di ogni giocatore di ogni partita e poi creare un altro dataset con le nuove partite . Per fare il dataset successivo vengono salvati i player 2 a 2 cioè prendo 2 match alla volta prendo i vincitori di queste 2 partite e creo un sola riga con questi 2 giocatori che corrisponde ad un altra partita.

“**net_dataset_player**”: ho creato questa funzione per copiare le features seguendo la stessa logica della funzione net_dataset cioè prendo 2 partite prendo i vincitori di queste due partite in base alla predizione fatta prima e prendo le feature di questi 2 giocatori e le metto in un dataset nuovo su una riga nuova.

“**TellMeTheWinner**”: ho creato questa funzione per poter stampare il vincitore del torneo perchè quando ho una sola riga sono certo di essere arrivato all'ultima partita e predico di nuovo il vincitore con la funzione sopra e in base a questa stampo semplicemente il vincitore prendendo il suo nome dal dataset creato in precedenza.

Considerazione Finale :

Dopo avere eseguito i 2 modelli posso dire con certezza che l’aggiunta delle nuove features ha migliorato tanto l’accuracy dei 2 modelli di classificazione. Per migliorare ancora il modello di previsione si potrebbero creare delle nuove features utilizzando dati esterni o si potrebbero creare altre funzioni con i dati del dataset che rispecchiano la bravura dei player.