Report dell'analisi "Determinazione dell'overall nel dataset dei giocatori di FIFA22"

Il dataset è stato scaricato dal sito Kaggle, e contiene le informazioni relative ai giocatori di calcio che compongono il videogioco FIFA22, il dataset presenta 19239 osservazioni e 110 variabili, ossia abbiamo 19239 giocatori i quali vengono descritti attraverso ben 110 variabili, le quali indicano ad esempio la nazionalità, la squadra di appartenenza, il ruolo, il valore economico del giocatore, la valutazione del giocatore in overall e potenziale ed anche ruolo per ruolo ed innumerevoli altre informazioni. L'obiettivo dell'analisi è andare a scoprire quali variabili determinano maggiormente la valutazione generale (*overall*) dei giocatori, o quale combinazione di variabili, la metodologia utilizzata è la regressione lineare proposta in tre diverse versioni.

Install and Import library

Le prime righe di codice sono, come da consuetudine, utilizzate per installare ed importare le librerie necessarie, possiamo osservare come quasi tutte le librerie importate provengano da da *pyspark.sql & pyspark.ml,* mentre le librerie *seaborn, matplotlib & pandas* sono state scaricate ed utilizzate esclusivamente per fini grafici.

Load Dataset, observing and drop first variables

Eseguo il load del dataset attraverso un semplice *spark.read.csv* ed osservo il dataset mediante un *count()* ed un *printSchema().*
E' sufficiente questa prima osservazione per poter eliminare attraverso *drop()* già 17 variabili reputate non utili ai fini dell'analisi in quanto relative ad i link per le immagini dei calciatori, delle magliette e dei loghi sia dei club che delle nazionali.

Changing economic values in K

Dato che il contesto dei calciatori professionisti presenta un'enorme variabilità in merito ai valori economici, basti pensare che all'interno di uno stesso campionato come quello italiano sono presenti sia calciatori che guadagnano 28000 (minimo salariale) e oltre 20000000, ho deciso di conseguenza di rappresentare tutti i valori economici, relativi alle variabili "Value; Wage; Relause_Clause" in multipli di K (ossia di 1000).
Per fare ciò ho utilizzato la funzione *withColumn()* e successivamente la funzione *drop()* per eliminare le vecchie variabili.

Remove Goalkeepers and Goalkeeper's variables

Siccome tutti i giocatori vengono descritti con le medesime variabili, pur essendo molto diversi in base al ruolo decido di eliminare i portieri e le caratteristiche relative ad i portieri dal dataset per avere delle osservazioni più omogenee riducendo il dataset ai giocatori di movimento.
Per essere più chiari alcune variabili come la velocità, il tiro, il dribbling, ma anche ad esempio l'età del singolo calciatore hanno ovviamente una significatività molto diversa se si sta descrivendo un giocatore di movimento o un portiere, la lentezza o l'età avanzata per un portiere non è necessariamente un limite, per un calciatore di corsa ovviamente si, ecco perché ho deciso di ridurre ai calciatori di movimento il dataset.

Anche qui è stato sufficiente utilizzare un *.where() & drop()*, il *describe* infine serve per visualizzare il cambiamento post eliminazione dei portieri.

Drop na from dataset

Per poter successivamente sviluppare i modelli ho la necessità di eliminare gli NA presenti nel dataset, attraverso un *ciclo for* identifico la presenza degli NA e la concentrazione nelle diverse variabili, da qui prendo la decisione di eliminare dal dataset le variabili *"player_tags ;nation_position; player_traits"* relative alla posizione preferita nella nazionale, e ai *tags & traits* ossia quelle etichette che vengono date ad alcuni calciatori come “funambolo, leggenda, cecchino” poiché presenti solo per meno del 10% dei calciatori presenti nel dataset.
Ho utilizzato in prima passata un *drop()* delle colonne sopra citate e in seconda battuta un semplice *dropna().*

Rappresentazioni grafiche

La parte relativa alle rappresentazioni grafiche ci mostra, grazie alle librerie *matplotlib, pandas e Seaborn* , in ordine di esecuzione nel notebook : due **istogrammi** incrociati che rappresentano il count per i valori di *potential & overall*, uno **scatterplot** che rappresenta l’interazione fra *wage, potential & international reputation* ; un altro **scatterplot** raffigurante l‘interazione fra *overall & wage ;* un altro **scatterplot** raffigurante l’interazione fra *international reputation & value* ; due distribuzioni dei valori di *value & wage ;* ed una **griglia** a 4 che rappresenta le distribuzioni di *wage , age , height & weight.*
Infine il grafico più importante è sicuramente il ***corrplot*** che ho scelto di eseguire filtrando come correlazione minima rappresentabile il valore 0.4 in modo tale da poter rendere più facilmente osservabile e di conseguenza comprensibile il grafico, anche se data la numerosità delle variabili risulta ancora non di facile comprensione.

Assembler-encoding e costruzione del primo modello

Attraverso l’osservazione del corrplot decido di costruire il primo modello di regressione lineare esclusivamente sulla variabile *movement_reactions* che presenta una correlazione di 0.88 con *overall.*
Procedo quindi con l’assembler, la suddivisione in train(0.7) e test(0.3) set, faccio girare il modello ed ottengo le seguenti conclusioni:

```
Risultati sul Test
RMSE = 3.31403
r2 = 0.762649
+-------+------------------+
|summary|           overall|
+-------+------------------+
|  count|              4822|
|   mean| 65.85420987142264|
| stddev|6.8030720993935745|
|    min|                47|
|    max|                92|
+-------+------------------+
```

```
Risultati sul Train
RMSE: 3.281636
r2: 0.769728
+-------+------------------+
|summary|           overall|
+-------+------------------+
|  count|             11198|
|   mean| 65.88337203071977|
| stddev|6.8389509274179705|
|    min|                47|
|    max|                93|
+-------+------------------+
```

Assembler-encoding e costruzione del secondo modello

Per la creazione del secondo modello ho deciso di utilizzare, per determinare l'*overall,* tutte le variabili relative alle valutazioni dei calciatori nei singoli ruoli.
Qui ho incontrato una difficoltà in quanto questi singoli variabili erano codificanti con il valore numerico dato dal gioco nella prima versione (settembre 2022) con una variabilità + o - 3 data dall'aggiornamento della settimana in cui è stato scaricato il dataset (3° di aprile 2022) ed inoltre queste variabili erano codificate come di tipo stringa.
Attaverso un *ciclo for* una ***regex*** *, il cast* ed un successivo drop sono riuscito a risolvere la problematica e dopo aver proceduto con l'*assembler* e la creazione del *train e test* set ho ottenuto dalla mia regressione lineare i seguenti risultati:

```
Risultati sul Train
RMSE: 3.137135
r2: 0.789561
```

summary	overall
count	11198
mean	65.88337203071977
stddev	6.8389509274179705
min	47
max	93

```
Risultati sul Test
RMSE = 3.14278
r2 = 0.786545
```

summary	overall
count	4822
mean	65.85420987142264
stddev	6.8030720993935745
min	47
max	92

Assembler-encoding-scaling e costruzione del terzo modello

Per la creazione del terzo modello invece ho deciso di utilizzare quasi la totalità delle variabili rimaste dopo il *data cleaning* in modo da poter confrontare i risultati con i precedenti modelli e comprendere meglio la capacità dei primi due modelli.
La complessità nella creazione di questo modello è relativa alla fase di *assembling-encoding-scaling* qui infatti mi sono servito di 2 ***assembler*** *questo perché la variabile work_rate* è passata prima dallo **string indexer** e successivamente dal **one hot encoder,** poi ho proceduto con il **robust_scaler,** tutte queste operazioni sono state eseguite nella medesima sequenza grazie all'utilizzo di una **pipeline.**
Infine come per i precedenti modelli ho creato il train e test set ed ho ottenuto i seguenti risultati:

```
Risultati sul Train          Risultati sul Test
RMSE: 1.650606               RMSE = 1.66817
r2: 0.941743                 r2 = 0.939861
+-------+------------------+ +-------+------------------+
|summary|           overall| |summary|           overall|
+-------+------------------+ +-------+------------------+
|  count|             11198| |  count|              4822|
|   mean| 65.88337203071977| |   mean| 65.85420987142264|
| stddev|6.8389509274179705| | stddev|6.8030720993935745|
|    min|                47| |    min|                47|
|    max|                93| |    max|                92|
+-------+------------------+ +-------+------------------+
```