**Progetto Big Data**

Confronto statistiche squadre di Serie A 2023/2024

STUDENTI:

**Gianmarco Brizzi**

**Simone Masciarelli**

**INTRODUZIONE**

**Descrizione della Tematica Scelta**

La presente analisi si concentra sul campionato di calcio di Serie A della stagione 2023-2024, una competizione di grande rilevanza in Italia e in Europa. L'obiettivo principale è quello di esaminare le performance delle squadre, in particolare della SS Lazio e dell'AS Roma, le due principali squadre di calcio della capitale italiana. Attraverso l'analisi dei risultati delle partite, si intende valutare le statistiche relative a vittorie, sconfitte, pareggi, gol segnati e gol subiti, per comprendere le dinamiche e le prestazioni delle squadre nel corso della stagione.  
Questo codice è stato incentrato sul confronto tra le due squadre della capitale ma potrebbe tranquillamente essere utilizzato cambiando pochi parametri per qualsiasi coppia di squadre.

**Presentazione del Dataset Utilizzato**

Per l'analisi, è stato utilizzato un dataset in formato JSON contenente i dati delle partite della Serie A per la stagione 2023-2024. Il dataset include informazioni dettagliate su ogni partita, tra cui:

* Data della partita
* Turno del campionato
* Squadre partecipanti (team1 e team2)
* Risultati finali (gol segnati da entrambe le squadre)

La struttura del dataset è stata esplorata attraverso il seguente schema, che fornisce una panoramica dei dati disponibili:

root

|-- matches: array (nullable = true)

| |-- element: struct (contains Null = true)

| | |-- date: string (nullable = true)

| | |-- round: string (nullable = true)

| | |-- score: struct (nullable = true)

| | | |-- ft: array (nullable = true)

| | | | |-- element: long (contains Null = true)

| | | |-- hit: array (nullable = true)

| | | | |-- element: long (containsNull = true)

| | |-- team1: string (nullable = true)

| | |-- team2: string (nullable = true)

| | |-- time: string (nullable = true)

|-- name: string (nullable = true)

**Obiettivi dell'Analisi**

Gli obiettivi principali dell'analisi sono i seguenti:

1. **Valutare le Performance**: Calcolare e confrontare le statistiche di prestazione delle squadre SS Lazio e AS Roma, analizzando il numero di vittorie, sconfitte, pareggi, gol segnati e gol subiti.
2. **Identificare le Tendenze**: Esaminare le tendenze nelle performance delle squadre durante il campionato, per individuare eventuali punti di forza e debolezza, analizzare le strategie di gioco e le aree di miglioramento per le squadre coinvolte.

Con questi obiettivi, l'analisi fornirà una visione delle dinamiche del campionato di Serie A e contribuirà a una comprensione più chiara delle prestazioni delle squadre che si vogliono analizzare all’interno della competizione.

**PREPARAZIONE DEI DATI**

**Caricamento e Pulizia dei Dati**

In questa fase, abbiamo iniziato caricando il dataset contenente i dati delle partite della Serie A per la stagione 2023-2024. Utilizziamo **PySpark** per gestire grandi volumi di dati in modo efficiente.

Il codice per il caricamento del file JSON è il seguente:

# Specifica il percorso del file JSON contenente i dati delle partite  
json\_file\_path = "C:\\Users\\gianm\\OneDrive\\Desktop\\it.1.json"  
  
# Carica i dati dal file JSON in un DataFrame  
partite\_df = spark.read.option("multiLine", "true").json(json\_file\_path)  
  
# Mostra la struttura del DataFrame per comprendere i tipi di dati e le colonne disponibili  
partite\_df.printSchema()

Il file JSON è stato caricato utilizzando spark.read.json, con l'opzione multiLine impostata su true per gestire correttamente i file JSON formattati su più righe. Successivamente, abbiamo utilizzato printSchema() per visualizzare la struttura del DataFrame e i tipi di dati delle colonne.

**Esplorazione Iniziale dei Dati Utilizzati**

L'output di printSchema() fornisce informazioni chiave sulla struttura del DataFrame, che include:

**root**

**|-- matches: array (nullable = true)**

**| |-- element: struct (containsNull = true)**

**| | |-- date: string (nullable = true)**

**| | |-- round: string (nullable = true)**

**| | |-- score: struct (nullable = true)**

**| | | |-- ft: array (nullable = true)**

**| | | | |-- element: long (containsNull = true)**

**| | | |-- ht: array (nullable = true)**

**| | | | |-- element: long (containsNull = true)**

**| | |-- team1: string (nullable = true)**

**| | |-- team2: string (nullable = true)**

**| | |-- time: string (nullable = true)**

**|-- name: string (nullable = true)**

Questo schema indica che il DataFrame principale ha una colonna denominata matches, che è un array di strutture. Ogni struttura rappresenta una partita e contiene i seguenti campi:

* date: la data della partita.
* round: il turno della partita.
* score: una struttura che include gli array ft (full-time) e ht (half-time), contenenti i punteggi delle squadre.
* team1: il nome della prima squadra.
* team2: il nome della seconda squadra.
* time: l'orario della partita.

Questa struttura ci permette di estrarre facilmente le informazioni necessarie per analizzare le prestazioni delle squadre nel campionato.

Inoltre, abbiamo utilizzato la funzione explode per trasformare l'array di partite in un DataFrame più accessibile:

# Estrai i dati delle partite in un nuovo DataFrame  
# Ogni riga rappresenta una partita con dettagli associati  
partite\_esplose\_df = partite\_df.select(explode("matches").alias("partita"))

Questa operazione ci consente di avere ogni partita come una riga separata nel nuovo DataFrame partite\_esplose\_df, facilitando l'analisi delle singole partite e delle statistiche delle squadre.

**ANALISI DEI DATI**

**Analisi Principali Effettuate sui Dati**

Dopo aver preparato i dati, ci siamo concentrati sull'analisi delle prestazioni delle due squadre, SS Lazio e AS Roma, nella stagione 2023-2024. Abbiamo implementato una classe chiamata AnalisiSquadra per calcolare diverse statistiche:

* **Vittorie**: Il numero totale di partite vinte da ciascuna squadra.
* **Sconfitte**: Il numero totale di partite perse da ciascuna squadra.
* **Pareggi**: Il numero totale di partite pareggiate.
* **Gol Segnati**: Il numero totale di gol realizzati.
* **Gol Subiti**: Il numero totale di gol subiti.

Ecco come vengono calcolate le vittorie, le sconfitte, i pareggi, i gol segnati e i gol subiti:

class AnalisiSquadra:

def \_\_init\_\_(self, partite\_df, nome\_squadra):

**"""**

**Inizializza l'oggetto AnalisiSquadra.**

**Args:**

**partite\_df (DataFrame): DataFrame contenente le informazioni delle partite esplose.**

**nome\_squadra (str): Nome della squadra per cui si desiderano calcolare le statistiche.**

**"""**

self.partite\_df = partite\_df

self.nome\_squadra = nome\_squadra

def vittorie(self):

**"""**

**Calcola il numero di vittorie della squadra.**

**Returns:**

**int: Numero totale di vittorie della squadra.**

**"""**

**# Filtra le righe in cui la squadra è il team1 e ha segnato più gol di team2, oppure è il team2**

**# e ha segnato meno gol di team1. Conta il numero di righe che soddisfano queste condizioni.**

return self.partite\_df.filter(

((col("partita.team1") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) > col("partita.score.ft").getItem(1))) |

((col("partita.team2") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) < col("partita.score.ft").getItem(1)))

).count()

def sconfitte(self):

**"""**

**Calcola il numero di sconfitte della squadra.**

**Returns:**

**int: Numero totale di sconfitte della squadra.**

**"""**

**# Filtra le righe in cui la squadra è il team1 e ha segnato meno gol di team2, oppure è il team2**

**# e ha segnato più gol di team1. Conta il numero di righe che soddisfano queste condizioni.**

return self.partite\_df.filter(

((col("partita.team1") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) < col("partita.score.ft").getItem(1))) |

((col("partita.team2") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) > col("partita.score.ft").getItem(1)))

).count()

def pareggi(self):

**"""**

**Calcola il numero di pareggi della squadra.**

**Returns:**

**int: Numero totale di pareggi della squadra.**

**"""**

**# Filtra le righe in cui il punteggio finale (ft) per team1 e team2 è uguale,**

**# sia che la squadra sia team1 o team2. Conta le righe che soddisfano questa condizione.**

return self.partite\_df.filter(

((col("partita.team1") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) == col("partita.score.ft").getItem(1))) |

((col("partita.team2") == self.nome\_squadra) & (col("partita.score.ft").getItem(0) == col("partita.score.ft").getItem(1)))

).count()

def gol\_segnati(self):

**"""**

**Calcola il numero totale di gol segnati dalla squadra.**

**Returns:**

**int: Numero totale di gol segnati dalla squadra, sia in casa che in trasferta.**

**"""**

**# Calcola i gol segnati dalla squadra quando gioca in casa (team1) sommando tutti i gol nella prima posizione dell'array "ft".**

gol\_home = self.partite\_df.filter(col("partita.team1") == self.nome\_squadra) \

.agg(spark\_sum(col("partita.score.ft").getItem(0))).collect()[0][0] or 0

**# Calcola i gol segnati dalla squadra quando gioca in trasferta (team2) sommando tutti i gol nella seconda posizione dell'array "ft".**

gol\_away = self.partite\_df.filter(col("partita.team2") == self.nome\_squadra) \

.agg(spark\_sum(col("partita.score.ft").getItem(1))).collect()[0][0] or 0

**# Somma i gol segnati in casa e in trasferta per ottenere il totale.**

return gol\_home + gol\_away

def gol\_subiti(self):

**"""**

**Calcola il numero totale di gol subiti dalla squadra.**

**Returns:**

**int: Numero totale di gol subiti dalla squadra, sia in casa che in trasferta.**

**"""**

**# Calcola i gol subiti dalla squadra quando gioca in casa (team1), sommando i gol dell'avversario nella seconda posizione dell'array "ft".**

gol\_subiti\_home = self.partite\_df.filter(col("partita.team1") == self.nome\_squadra) \

.agg(spark\_sum(col("partita.score.ft").getItem(1))).collect()[0][0] or 0

**# Calcola i gol subiti dalla squadra quando gioca in trasferta (team2), sommando i gol dell'avversario nella prima posizione dell'array "ft".**

gol\_subiti\_away = self.partite\_df.filter(col("partita.team2") == self.nome\_squadra) \

.agg(spark\_sum(col("partita.score.ft").getItem(0))).collect()[0][0] or 0

**# Somma i gol subiti in casa e in trasferta per ottenere il totale.**

return gol\_subiti\_home + gol\_subiti\_away

Questa classe ci permette di calcolare le statistiche richieste per entrambe le squadre.

Dopo aver creato istanze di AnalisiSquadra per SS Lazio e AS Roma, abbiamo stampato le statistiche per ciascuna squadra:

# Stampa i risultati delle statistiche

print(f"Lazio: Vittorie: {analisi\_lazio.vittorie()}, Sconfitte: {analisi\_lazio.sconfitte()}, Pareggi: {analisi\_lazio.pareggi()}, Gol segnati: {analisi\_lazio.gol\_segnati()}, Gol subiti: {analisi\_lazio.gol\_subiti()}")

print(f"Roma: Vittorie: {analisi\_roma.vittorie()}, Sconfitte: {analisi\_roma.sconfitte()}, Pareggi: {analisi\_roma.pareggi()}, Gol segnati: {analisi\_roma.gol\_segnati()}, Gol subiti: {analisi\_roma.gol\_subiti()}")

**Visualizzazione e/o Interpretazione dei Dati**

Dopo aver calcolato le statistiche, abbiamo anche realizzato un confronto tra le due squadre, analizzando come si confrontano in termini di vittorie, sconfitte, pareggi, gol segnati e gol subiti.

L'output di confronto tra Lazio e Roma è il seguente:

Confronto tra Lazio e Roma:

SS Lazio e AS Roma hanno lo stesso numero di Vittorie.

SS Lazio ha 2 Sconfitte in più rispetto a AS Roma.

SS Lazio ha 2 Pareggi in meno rispetto a AS Roma.

AS Roma ha 16 Gol segnati in più rispetto a SS Lazio.

AS Roma ha 7 Gol subiti in più rispetto a SS Lazio.

Questa interpretazione delle statistiche consente di comprendere meglio le prestazioni di ciascuna squadra nella stagione. Ad esempio, possiamo notare che, nonostante Lazio e Roma abbiano lo stesso numero di vittorie, la Roma ha segnato significativamente più gol e ha una media di sconfitte inferiore. Questi dettagli offrono una panoramica chiara delle prestazioni delle squadre in relazione l'una all'altra.

**CONCLUSIONI**

**Riassunto dei Risultati Ottenuti**

L'analisi delle prestazioni di SS Lazio e AS Roma nella stagione 2023-2024 ha portato a diverse conclusioni significative. Attraverso l'implementazione della classe AnalisiSquadra, siamo stati in grado di calcolare vari indicatori delle performance di ciascuna squadra. Di seguito sono riportati i risultati principali:

* **Vittorie**: Entrambe le squadre hanno mostrato prestazioni simili in termini di vittorie, con SS Lazio e AS Roma che hanno collezionato lo stesso numero di partite vinte.
* **Sconfitte**: SS Lazio ha subito 2 sconfitte in più rispetto a AS Roma, suggerendo una leggera superiorità della Roma in termini di risultati negativi.
* **Pareggi**: La Lazio ha registrato 2 pareggi in meno rispetto alla Roma, indicando un approccio forse più aggressivo nella ricerca di vittorie.
* **Gol Segnati**: AS Roma ha segnato 16 gol in più rispetto alla Lazio, il che evidenzia una maggiore capacità offensiva da parte della Roma.
* **Gol Subiti**: Tuttavia, Roma ha subito 7 gol in più rispetto alla Lazio, suggerendo una difesa meno solida.

Queste statistiche forniscono un quadro dettagliato delle prestazioni di entrambe le squadre, evidenziando le aree in cui ciascuna eccelle e quelle in cui potrebbe migliorare.

**Considerazioni Finali e Possibili Sviluppi Futuri**

L'analisi ha dimostrato l'efficacia dell'utilizzo di PySpark per gestire e analizzare dati sportivi, grazie alla sua capacità di elaborare grandi volumi di informazioni in modo rapido ed efficiente. Le statistiche ottenute possono essere utilizzate per ulteriori ricerche e analisi, ad esempio:

* **Analisi Approfondite**: Approfondire ulteriormente le performance individuali dei giocatori, esaminando statistiche come gol, assist, ammonizioni e infortuni per comprendere l'impatto di ciascun giocatore sulla performance della squadra.
* **Visualizzazione dei Dati**: Implementare strumenti di visualizzazione dei dati, come grafici e dashboard, per rendere più comprensibili e accessibili i risultati ottenuti.

In conclusione, questa analisi delle partite della Serie A ci ha mostrato come approcciare l’analisi dei dati sportivi, dimostrando come i dati possano offrire preziose informazioni sulle performance delle squadre e guidare le decisioni strategiche nel calcio.