

Homework 1: Apache Spark



Studenti: Antimo Barbato
Matricola: M63/1079

1: Introduzione ad Apache Spark

Apache Spark è un potente motore di elaborazione dati open-source progettato per velocizzare le analisi su larga scala. Nato nei laboratori AMPLab dell'Università di Berkeley nel 2009 e successivamente adottato dalla Apache Software Foundation nel 2013, Spark è diventato uno strumento fondamentale per il trattamento dei big data.

Spark si distingue per la sua capacità di elaborare dati in memoria, il che lo rende significativamente più veloce rispetto ai tradizionali sistemi basati su disco come Hadoop MapReduce. La sua architettura è costruita attorno ai RDD (Resilient Distributed Datasets), che sono dataset distribuiti, immutabili e tolleranti agli errori, permettendo operazioni parallele efficienti su grandi cluster di computer.

L'ecosistema di Apache Spark è composto da diversi componenti chiave:

- **Spark Core:** Il cuore del motore di elaborazione, responsabile delle funzionalità di I/O di base e dell'esecuzione distribuita.
- **Spark SQL:** Permette di eseguire query su dati strutturati utilizzando SQL o HQL.
- **Spark Streaming:** Consente l'elaborazione di dati in streaming in tempo reale.
- **MLlib:** Una libreria scalabile per il machine learning, che include algoritmi per la classificazione, regressione e clustering.
- **GraphX:** Un framework per l'elaborazione di grafi e analisi di rete.

Grazie all

a sua versatilità e potenza, Apache Spark è utilizzato in una vasta gamma di applicazioni, dalla data science all'analisi in tempo reale, rendendolo uno strumento indispensabile per le aziende che gestiscono grandi volumi di dati.

2: Dataset – Videogames Sales

Il dataset "Video Games Sales" (formato csv) offre una panoramica dettagliata delle vendite globali di videogiochi, includendo informazioni come il titolo del gioco, la piattaforma, il genere, l'editore, le vendite regionali (Nord America, Europa, Giappone e Resto del Mondo), le vendite globali e le recensioni. Questo dataset è utile per analizzare le tendenze del mercato dei videogiochi, confrontare le performance degli editori e studiare le preferenze dei giocatori in diverse regioni.

Vediamo nel dettaglio gli attributi relativi al dataset

- **Rank:** La classifica del videogioco basata sul volume di vendite globali.
- **Game Title:** Il nome del videogioco.
- **Platform:** La piattaforma su cui è disponibile il gioco, come PC, PS4, Xbox One, ecc.
- **Year:** L'anno in cui il gioco è stato rilasciato.
- **Genre:** Il genere del gioco, come azione, avventura, corse, ecc.
- **Publisher:** La società che ha pubblicato il gioco.
- **North America:** Il numero di unità vendute in Nord America, in milioni.
- **Europe:** Il numero di unità vendute in Europa, in milioni.
- **Japan:** Il numero di unità vendute in Giappone, in milioni.
- **Rest of World:** Il numero di unità vendute nel resto del mondo, esclusi Nord America, Europa e Giappone, in milioni.
- **Global:** Il numero totale di unità vendute in tutto il mondo, in milioni.
- **Review:** Il punteggio delle recensioni del gioco, su una scala da 1 a 10.

2.1: Piattaforma di analisi dei dati: Databricks

La piattaforma utilizzata per analizzare il dataset è Databricks. Databricks è una piattaforma di analisi dei dati basata su cloud che facilita l'elaborazione e l'analisi di grandi volumi di dati. Fondata dai creatori di Apache Spark, Databricks offre un ambiente integrato per la gestione dei dati, l'analisi e il machine learning.

Caratteristiche principali di Databricks:

- **Integrazione con Apache Spark:** Databricks sfrutta la potenza di Apache Spark per l'elaborazione distribuita dei dati, consentendo analisi rapide e scalabili.
- **Notebook Collaborativi:** Gli utenti possono creare e condividere notebook interattivi per scrivere codice, eseguire query e visualizzare i risultati in tempo reale.
- **Gestione dei Cluster:** Databricks semplifica la creazione e la gestione dei cluster, permettendo di scalare facilmente le risorse computazionali in base alle esigenze.
- **Supporto Multi-Linguaggio:** La piattaforma supporta diversi linguaggi di programmazione, tra cui Python, SQL, Scala e R, rendendola versatile per vari tipi di analisi.

Nello specifico per analizzare il dataset, si è anzitutto creato un cluster.

Compute

antimo barbato's Cluster •

Configurazione Notebook (1) Librerie Log eventi UI Spark Log driver Metriche App UI compute Spark - Master ▾

Versione Runtime Databricks

12.2 LTS (includes Apache Spark 3.3.2, Scala 2.12)

Tipo di driver

Community Optimized Memoria di 15,3 GB e 2 Cores

Istanza

Memoria da 15 GB gratuita: in qualità di utente della Community Edition, il compute terminerà automaticamente dopo un periodo di inattività di una o due ore. Per maggiori opzioni di configurazione [clicca qui](#), esegui l'upgrade dell'abbonamento a Databricks. [clicca qui](#)

Spark JDBC/ODBC

Configurazione Spark ⓘ

spark.databricks.rocksDB.fileManager.useCommitService false

Variabili di ambiente ⓘ

PYSPARK_PYTHON=/databricks/python3/bin/python3

Successivamente si è caricato il dataset

default.videogames_2_csv Aggiorna

antimo barbato's Cluster ▾

Schema:

	col_name	data_type	comment
1	index	int	(null)
2	Rank	int	(null)
3	Game Title	string	(null)
4	Platform	string	(null)
5	Year	double	(null)
6	Genre	string	(null)
7	Publisher	string	(null)
8	North America	double	(null)
9	Europe	double	(null)
10	Japan	double	(null)
11	Rest of World	double	(null)
12	Global	double	(null)
13	Review	double	(null)

Dati campione:

	index	Rank	Game Title	Platform	Year	Genre	Publisher	1.2 North America	1.2 Europe	1.2 Japan
1	0	1	Wii Sports	Wii	2006	Sports	Nintendo	40.43	28.39	3.77
2	1	2	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81
3	2	3	Mario Kart Wii	Wii	2008	Racing	Nintendo	14.5	12.22	3.83
4	3	4	Wii Sports Resort	Wii	2009	Sports	Nintendo	14.82	10.51	2.18
5	4	5	Tetris	GB	1989	Puzzle	Nintendo	23.2	2.26	4.22
6	5	6	New Super Mario Bros.	DS	2006	Platform	Nintendo	10.85	8.87	6.48
7	6	7	Wii Play	Wii	2006	Misc	Nintendo	13.83	9.11	2.93
8	7	8	Duck Hunt	NES	1984	Shooter	Nintendo	26.93	0.63	0.28
9	8	9	New Super Mario Bros. Wii	Wii	2009	Platform	Nintendo	13.35	6.48	4.66
10	9	10	Nintendogs	DS	2005	Simulation	Nintendo	9.02	10.81	1.93
11	10	11	Drilldowns Code / Cluster Version	GB	1990	Role-Playing	Nintendo	0	4.18	7.3

2.2: Pre-Processing con PySpark

Durante il pre-processing dei dati, sono stati gestiti i valori nulli procedendo all'eliminazione per righe.

Durante tale fase si è deciso di non eliminare alcuna colonna, poiché ritenute tutte fondamentali ai fini dell'analisi

1) Carico il dataset da DBFS e lo visualizzo

```
#Carico il dataset da DBFS
df = spark.read.csv('/FileStore/tables/videogames-2.csv', header=True,
inferSchema=True)

display(df.limit(4))
```

index	Rank	Game Title	Platform	Year	Genre	Publisher	North America	Europe	Japan	Rest of World	Global	Review
0	1	Wii Sports	Wii	2006.00	Sports	Nintendo						
1	2	Super Mario Bros.	NES	1985.00	Platform	Nintendo						
2	3	Mario Kart Wii	Wii	2008.00	Racing	Nintendo						
3	4	Wii Sports Resort	Wii	2009.00	Sports	Nintendo						

2) Controllo il numero di valori nulli nel dataset

```
from pyspark.sql.functions import col, count, when

# Conta i valori NULL in ogni colonna
null_counts = df.select([count(when(col(c).isNull(), c)).alias(c) for c in
df.columns])

# Conta il numero totale di valori per ogni colonna
total_counts = df.select([count(col(c)).alias(c) for c in df.columns])

# Mostra il conteggio dei valori nulli e totali per ogni colonna
total_counts.show()
null_counts.show()
```

index	Rank	Game Title	Platform	Year	Genre	Publisher	North America	Europe	Japan	Rest of World	Global	Review
1907	1907	1907	1907	1878	1907	1905	1907	1907	1907	1907	1907	1907

index	Rank	Game Title	Platform	Year	Genre	Publisher	North America	Europe	Japan	Rest of World	Global	Review
0	0	0	0	29	0	2	0	0	0	0	0	0

3) Elimino i valori nulli dal dataset

```
# Elimina le righe con valori nulli
df_cleaned = df.dropna()

# Mostra il DataFrame pulito
df_cleaned.show()
```

A seguito vengono effettuate una serie di query per analizzare il dataset

2.3: Query con PySpark

Query 1: Conteggio dei giochi con lo stesso nome e ordinati in ordine decrescente

```
from pyspark.sql.functions import col

# Raggruppa i dati per Game Title del gioco e conta
conteggio_giochi = df_cleaned.groupBy("Game Title").count()

# Ordina i risultati in ordine decrescente in base al conteggio
giochi_ord = conteggio_giochi.orderBy(col("count").desc())

# Mostra i risultati ordinati
display(giochi_ord)
```

	Game Title	count
1	FIFA Soccer 08	6
2	LEGO Indiana Jones: The Original Adventures	5
3	The Simpsons Game	5
4	Star Wars: The Force Unleashed	5
5	WWE SmackDown vs Raw 2008	5
6	Pro Evolution Soccer 2008	5
7	FIFA Soccer 10	5
8	LEGO Star Wars: The Complete Saga	4
9	Madden NFL 08	4
10	The Sims 3	4
11	LEGO Star Wars: The Video Game	4
12	Need for Speed Carbon	4
13	Spider-Man 2	4
14	LEGO Batman: The Videogame	4
15	Spider-Man: The Movie	4

Query 2: Media vendite globali ordinate in ordine decrescente

```
from pyspark.sql.functions import col, avg

# media vendite globali per genere
media_genere =
df_cleaned.groupBy("Genre").agg(avg("Global").alias("Avg_Global_Sales"))

# ordina in ordine decrescente
media_gen_ord = media_genere.orderBy(col("Avg_Global_Sales").desc())

# Mostra i risultati ordinati
display(media_gen_ord.limit(6))
```

	A _C Genre	1.2 Avg_Global_Sales
1	Platform	3.18967741935484
2	Role-Playing	2.889122807017545
3	Misc	2.7264705882352938
4	Shooter	2.7136274509803946
5	Puzzle	2.469318181818182
6	Racing	2.438804347826087

Concentriamoci sui primi 3 per la prossima query per vedere quale Publisher ne ha prodotti maggiormente

Query 3: Filtriamo in base ai 3 generi, raggruppiamo in base al publisher e contiamo il numero di giochi prodotti per genere con relativo ordinamento decrescente

```
from pyspark.sql.functions import col

# Filtra i giochi per i generi "Platform", "Role-Playing" e "Misc"
filtered_games = df_cleaned.filter(col("Genre").isin("Platform", "Role-Playing",
"Misc"))

# Raggruppa i dati per publisher e genere, quindi conta il numero di giochi per
ciascun publisher e genere
conteggio_publisher = filtered_games.groupBy("Publisher", "Genre").count()

# Ordina i risultati in ordine decrescente in base al conteggio
conteggio_publisher_ord = conteggio_publisher.orderBy(col("count").desc())

# Mostra i risultati ordinati
display(conteggio_publisher_ord.limit(10))
```

	A _C Publisher	A _C Genre	1.2 count
1	Nintendo	Platform	67
2	Nintendo	Role-Playing	38
3	Nintendo	Misc	32
4	Sega	Platform	30
5	Sony Computer Entertainme...	Misc	27
6	Square Enix	Role-Playing	23
7	Activision	Misc	23
8	Sony Computer Entertainme...	Platform	21
9	Square	Role-Playing	16
10	THQ	Platform	16

Continuiamo analizzando a questo punto le vendite globali di Nintendo per il genere Platform

**Query 4: Filtro per il genere "Platform" e Publisher "Nintendo".
Calcolo la media delle vendite globali per il publisher Nintendo**

```
from pyspark.sql.functions import col, avg

# Filtra i giochi di genere "Platform" pubblicati da Nintendo
nintendo_platform = df_cleaned.filter((col("Publisher") == "Nintendo") &
                                       (col("Genre") == "Platform"))

# Calcola la media delle vendite globali per i giochi di genere "Platform"
# pubblicati da Nintendo
avg_nintendo_platform =
nintendo_platform.agg(avg("Global").alias("Avg_Global_Sales"))

# Mostra la media delle vendite globali
display(avg_nintendo_platform)
```

	1.2 Avg_Global_Sales
1	5.740597014925373

Successivamente calcolo la media delle vendite annue di nintendo

Query 5: Filtro per il Publisher "Nintendo" e calcolo la media di vendite globali annue di Nintendo

```
from pyspark.sql.functions import col, avg

# Filtra i giochi pubblicati da Nintendo
nintendo_games = df_cleaned.filter(col("Publisher") == "Nintendo")

# Raggruppa i dati per anno e calcola la media delle vendite globali per ciascun
# anno
avg_vendite_nintendo =
nintendo_games.groupBy("Year").agg(avg("Global").alias("AVG_Annuale"))

# Ordina i risultati in ordine decrescente in base alla media delle vendite
# globali annue
avg_vendite_nintendo_ord =
avg_vendite_nintendo.orderBy(col("AVG_Annuale").desc())

# Mostra i risultati ordinati
display(avg_vendite_nintendo_ord.limit(5))
```

	1.2 Year	1.2 AVG_Annuale
1	2006	11.232352941176472
2	1989	9.833333333333334
3	1985	9.704
4	2009	8.651428571428571
5	1988	6.860000000000001

Query 6: Filtro in base a Review e Global per ottenere i giochi con valutazione maggiore di 8 e con vendite superiori a 20 milioni

#Giochi con una valutazione superiore all'8 e vendite globali superiori a 20milioni

```
df_cleaned.filter((df_cleaned["Review"] > 8) & (df_cleaned["Global"] > 20)) \
.select("Game Title", "Review", "Global") \
.orderBy(col("Global").desc()) \
.show()
```

Game Title	Review	Global
Wii Sports	76.28	81.12
Super Mario Bros.	91.0	40.24
Mario Kart Wii	82.07	33.55
Wii Sports Resort	82.65	31.52
Tetris	88.0	30.26
New Super Mario B...	90.0	29.08
Wii Play	61.64	28.71
Duck Hunt	84.0	28.31
New Super Mario B...	88.18	26.75
Nintendogs	85.0	24.5
Pokémon Gold / Si...	89.0	23.1
Wii Fit	81.2	22.74
Mario Kart DS	91.34	22.47
Wii Fit Plus	80.83	21.15
Grand Theft Auto:...	95.08	20.81
Super Mario World	94.0	20.61
Brain Age: Train ...	78.05	20.02

A seguito di tale risultato si è deciso di inserire nella select anche l'attributo relativo all'anno, per verificare il primo gioco a che anno fa riferimento, considerando che nintendo ha le migliori vendite medie annue nel 2006

Game Title	Review	Global	Year
Wii Sports	76.28	81.12	2006.0
Super Mario Bros.	91.0	40.24	1985.0
Mario Kart Wii	82.07	33.55	2008.0
Wii Sports Resort	82.65	31.52	2009.0
Tetris	88.0	30.26	1989.0
New Super Mario B...	90.0	29.08	2006.0
Wii Play	61.64	28.71	2006.0
Duck Hunt	84.0	28.31	1984.0
New Super Mario B...	88.18	26.75	2009.0
Nintendogs	85.0	24.5	2005.0

Come si può notare nella top 10 rientrano 3 giochi Nintendo con un numero di vendite globali pari a 81 milioni, 29 milioni e 28 milioni per quanto riguarda l'anno 2006.

A questo punto può essere interessante vedere tutti i giochi venduti da nintendo nel 2006.

Query 7: Filtro per Publisher e Anno.

Ordino in base alle vendite globali per ottenere una lista dei giochi più venduti di nintendo nel 2006

```
from pyspark.sql.functions import col

# Filtra i giochi pubblicati da Nintendo nel 2006
nintendo_games_2006 = df_cleaned.filter((col("Publisher") == "Nintendo") &
(col("Year") == 2006))

# Ordina i giochi in ordine decrescente in base alle vendite globali
nintendo_games_2006_ord = nintendo_games_2006.orderBy(col("Global").desc())

# Seleziona i titoli dei giochi e le vendite globali
stampa_nintendo = nintendo_games_2006_ord.select("Game Title", "Global")

display(stampa_nintendo)
```

	A ^B Game Title	1.2 Global
1	Wii Sports	81.12
2	New Super Mario Bros.	29.08
3	Wii Play	28.71
4	Pokémon Diamond / Pearl Version	18.05
5	The Legend of Zelda: Twilight Princess	6.76
6	Clubhouse Games	3.35
7	English Training: Have Fun Improving Your Skills!	3.33
8	Personal Trainer: Cooking	3.09
9	WarioWare: Smooth Moves	2.84
10	Yoshi's Island DS	2.69
11	Pokémon Ranger	2.11
12	Tetris DS	2.07
13	Kirby Squeak Squad	1.91
14	The Legend of Zelda: Twilight Princess	1.59
15	Mario Hoops 3 on 3	1.57

Query 8: Raggruppamento in base a platform e year e effettuo il conteggio per ottenere il numero di giochi in base all'anno.

```
#Contare i giochi per piattaforme e anno di rilascio
df_cleaned.groupBy("Platform",
"Year").count().orderBy(col("count").desc()).show()
```

Platform	Year	count
PS2	2002.0	66
PS2	2004.0	64
PS2	2003.0	59
PS	1998.0	55
PS2	2005.0	48
PS	1999.0	45
PS2	2001.0	43
PS3	2008.0	43
X360	2010.0	43
X360	2008.0	42
wii	2008.0	40
PS	1997.0	38
DS	2007.0	37
PS3	2010.0	37
wii	2007.0	36
X360	2011.0	36
DS	2008.0	35
PS3	2011.0	34

Query 9: Raggruppamento per Publisher e calcolo della somma relative alle vendite sia regionali che globali e ordinamento in base alle globali

```
from pyspark.sql.functions import col, sum

# Raggruppa i dati per publisher e calcola la somma delle vendite per ciascuna regione
vendite_tot_publisher = df_cleaned.groupBy("Publisher").agg(
    sum("North America").alias("Vendite_NA"),
    sum("Europe").alias("Vendite_EU"),
    sum("Japan").alias("Vendite_JP"),
    sum("Rest of World").alias("Vendite_RW"),
    sum("Global").alias("Vendite_Globali")
)

# Ordina i risultati in ordine decrescente in base alla somma delle vendite globali
vendite_tot_publisher_ord =
vendite_tot_publisher.orderBy(col("Vendite_Globali").desc())

# Mostra i risultati ordinati
display(vendite_tot_publisher)
```

	^B Publisher	1.2 Vendite_NA	1.2 Vendite_EU	1.2 Vendite_JP	1.2 Vendite_RW	1.2 Vendite_Globali
1	Nintendo	687.7900000000001	341.49000000000046	338.0399999999997	80.53999999999998	1447.8100000000013
2	Electronic Arts	345.24999999999966	203.04000000000008	8.719999999999985	67.23000000000008	624.18
3	Sony Computer Entertainment	167.58999999999992	119.19	50.27000000000001	40.51000000000004	377.60999999999996
4	Activision	218.67999999999995	112.82000000000002	3.659999999999953	34.85000000000002	369.97999999999996
5	Take-Two Interactive	112.63999999999999	67.43	3.819999999999963	24.499999999999996	208.41999999999999
6	Ubisoft	103.22999999999996	69.61	1.7900000000000001	21.690000000000001	196.31999999999994
7	Microsoft Game Studios	110.68000000000002	43.220000000000006	2.3700000000000006	13.369999999999992	169.73000000000005
8	THQ	80.19000000000004	47.00000000000002	2.579999999999987	13.269999999999996	142.98
9	Sega	60.530000000000015	39.73000000000001	11.259999999999998	10.269999999999994	121.8
10	Capcom	49.41999999999999	24.310000000000013	32.91000000000001	7.76	114.33000000000001
11	Konami Digital Entertainment	30.41	43.57	21.710000000000008	12.009999999999998	107.66999999999999
12	Namco Bandai Games	30.279999999999994	17.040000000000003	18.750000000000004	5.599999999999997	71.68999999999998
13	Square Enix	21.930000000000003	12.019999999999998	24.379999999999995	5.08	63.410000000000004
14	LucasArts	33.739999999999995	18.53	0.200000000000000...	6.439999999999995	58.88999999999999
15	Eidos Interactive	27.52	19.680000000000003	2.779999999999994	4.04	54.01999999999999