Benchmarking e sentiment analysis su cluster spark

Progetto per il corso: Large scale data management

Autore: Lorenzo Colombo – Mat. 885895

Scenario e obiettivi

Analisi del sentiment di un dataset massivo eterogeneo composto da recensioni utente su venti applicazioni distinte, estratte dal Google Play Store

Le recensioni provengono da 20 file csv e riguardano le top 20 applicazioni presenti sul Play Store riportando informazioni su contenuto della recensione e punteggio

progettare, implementare e testare una pipeline di Big Data Analytics utilizzando Apache Spark in un ambiente distribuito su Cloud Azure effettuando benchmark su prestazioni e scalabilità

Architettura della soluzione

Infrastruttura Cloud:

Ambiente: Microsoft Azure

Nodi: 5 VM totali

- **1 Master:** Coordinamento e gestione del cluster
- **4 Workers:** Nodi computazionali dedicati ai task paralleli

Rete: Virtual Network privata per garantire bassa latenza tra i nodi

Stack software:

Motore di calcolo: Apache Spark 3.5.0

Storage: Apache Hadoop 3.3.6

Linguaggio: Python per la definizione della pipeline NLP e la data visuallization.

Strategia di Storage:

HDFS Replication Factor: 3

Data Locality: Spark assegna i task ai worker che ospitano fisicamente i blocchi di dati, riducendo il traffico di rete

Formato Dati: Apache Parquet

Data Engineering e Preparazione

Ingestione Dati

- **Input:** 20 File CSV eterogenei
- **Action:** Unificazione schema e normalizzazione
- **Arricchimento:** aggiunta della sorgente `app_name` per tracciabilità

Data Augmentation

- **Obiettivo:** aumentare il volume dei dati per effettuare stress test
- **Tecnica:** Funzioni Spark `explode` e `array_repeat`.
- **Integrità:** Generazione distribuita di chiave primaria `unique_id`.

Ottimizzazione e salvataggio dati

- **Output:** Scrittura su HDFS in formato parquet.
- **Vantaggio:** conversione in un formato più adatto a questo tipo di analisi

Pipeline di analisi distribuita

Pre-processing e Cleaning

- **Obiettivo:** Riduzione del rumore
- **Tecnica:** Utilizzo di Regex per eliminare caratteri non alfanumerici, punteggiatura e normalizzazione del testo
- **Risultato:** Testo uniforme pronto per la tokenizzazione

Stopwords Removal

- **Filtraggio base:** Rimozione di articoli, proposizioni e congiunzioni
- **Filtraggio di contesto:** Rimozione termini ad alta frequenza ma poco informativi specifici del contesto.
- **Output:** Solo coppie di parole utili

Estrazione dei bigrammi

- **Approccio:** suddividere il testo delle recensioni in coppie di parole
- **Motivazione:** Preservare il contesto semantico

Aggregazione MapReduce

- **Calcolo:** Conteggio distribuito delle occorrenze delle coppie di parole
- **Salvataggio ed esportazione:** esportazione dei risultati in formato csv

Benchmarking e Scalabilità

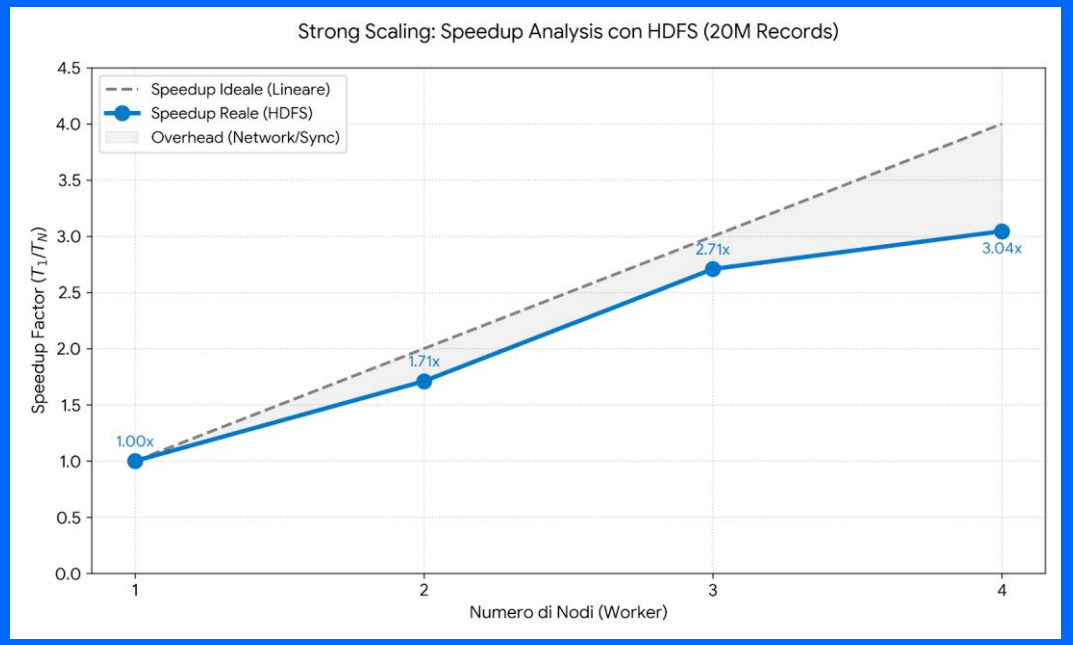
Metodologia di Test:

- Volume dati fisso vs Risorse variabili
- Esecuzione della stessa pipeline NLP incrementando i nodi worker da 1 a 4
- Misurazione del tempo di esecuzione

Analisi dell'Efficienza:

- **Sweet Spot a 3 nodi:** Efficienza massima raggiunta (**90%**)
- **Punto di Saturazione con 4 nodi:** L'efficienza scende al **76%.**

Config.	Core Totali	RAM	Load (s)	Calcolo (s)	Totale (s)
1 Nodo	4 CPU	10 GB	5.19	237.54	243.96
2 Nodi	8 CPU	20GB	5.78	135.75	142.63
3 Nodi	12 CPU	30 GB	5.78	83.2	90.08
4 Nodi	16 CPU	40 GB	5.66	73.40	80.12



Analisi del sentiment

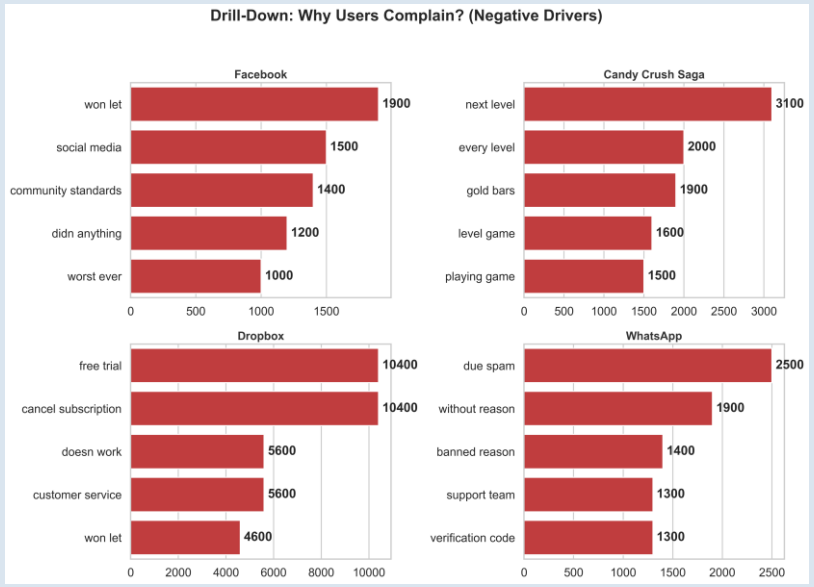
Livello Macro: Global Ecosystem Analysis

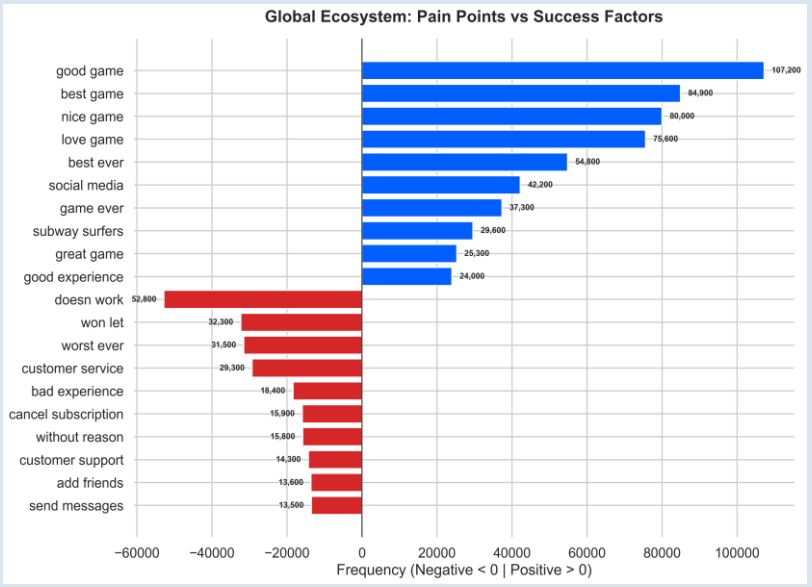
- **Approccio:** Analisi dell'intero dataset come unico corpus così da evidenziare le tendenze trasversali
- **Risultato:** Identificazione di 3 cluster di criticità universali:
  - **Technical Failure:** la stabilità tecnica pesa per oltre il 40% sulle recensioni negative, indipendentemente dal tipo di App.
  - **Customer Support Gap:** l'utente usa la recensione pubblica come canale per segnalare l'inefficacia dei mezzi di supporto tradizionali.
  - **Dark Patterns & Billing:** forte insofferenza verso rinnovi automatici poco trasparenti e gestione dei free trial.

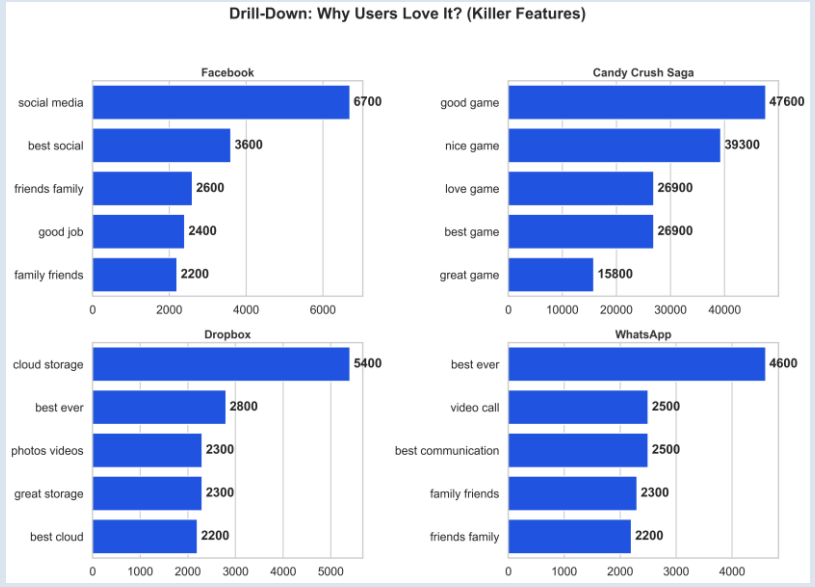
Livello Micro: Drill-Down Application Analysis

- **Tecnica:** Utilizzo delle window function per partizionare l'analisi per App
- **Risultato:** I driver di insoddisfazione cambiano radicalmente tra categorie (es. Gaming vs Utility), rendendo meno funzionali le strategie correttive generaliste tralasciando alcuni aspetti che riguardano il contesto dell’app.

Data visualization







Conclusioni

- **Solidità Infrastrutturale:** I benchmark confermano l'efficacia del cluster distribuito in quanto è stato ottenuta una scalabilità quasi lineare, dimostrando che il sistema converte efficientemente le risorse aggiuntive in velocità. L'integrazione con HDFS ha inoltre garantito la sopravvivenza del job anche in caso di guasti critici.
- **Valore Analitico:** è stata superata la superficialità delle metriche quantitative e, mediante l'analisi semantica granulare, è stato possibile diagnosticare le cause radice specifiche per ogni applicazione e generali di insoddisfazione, trasformando milioni di dati grezzi in indicazioni strategiche reali.
- **Verdetto Tecnologico:** lo stack tecnologico utilizzato si conferma una soluzione industriale reale e capace di abbattere drasticamente la latenza decisionale su grandi volumi di dati non strutturati.

Grazie

Colombo Lorenzo

Mat. 885895

l.colombo146@campus.unimib.it