A.D. 1308
unipg
DIPARTIMENTO
DI INGEGNERIA

Tesina di progetto

Data Intensive Application and Big Data

Corso di Laurea Magistrale in Ingegneria Informatica e Robotica
Curriculum Data Science
A.A. 2023-2024
Dipartimento di Ingegneria

docente
Prof. Fabrizio Montecchiani

Analisi delle performance di un cluster Hadoop per l'addestramento di un algoritmo di regressione lineare tramite Apache Spark

studente

363551 **Andrea Tomassoni** andrea.tomassoni@studenti.unipg.it

Data ultimo aggiornamento: 9 giugno 2024

0. Indice

1. Introduzione

Lo scopo del progetto è quello di andare a svolgere un analisi delle prestazioni di un cluster Hadoop sul quale viene eseguito l'addestramento di un algoritmo di machine learning per la regressione lineare.

1.1 Dataset Utilizzato

Per l'addestramento dell'algoritmo è stato utilizzato un dataset reperibile dalla piattaforma kaggle che contiene circa 850'000 record (55MB).
Ogni record è formato dai seguenti campi:

- Price: prezzo della vettura (utilizzata come label)
- Year: anno di acquisto della vettura
- Mileage: n° di km percorsi
- City: città in cui è stata acquistata
- State: stato in cui è stata acquistata
- Vin: identificativo della vettura
- Make: produttore del veicolo
- Model: modello del veicolo

Prima di essere passati all'algoritmo questi dati subiscono una serie di trasformazioni per renderli adatti allo scopo.
Nello specifico, è stata rimossa la colonna "vin", è stato fatto l'encoding (tramite one-hot-encoding) dei campi "city", "state", "make" e "model". Infine, tutti i campi sono stati normalizzati.

1.2 Cluster

Il cluster strutturato tramite hadoop è ospitato su tre macchine virtuali da 8 GB di RAM e 4 vCores su cui è installato Ubuntu Server 2024. Le tre macchine virtuali sono installate su tre macchine host collegate tra loro tramite uno switch Gigabit Ethernet.

2. Tecnologie utilizzate

2.1 Apache Hadoop

Per quanto riguarda la strutturazione del cluster, questa è stata fatta tramite tecnologia Hadoop e nello specifico sono state utilizzate le componenti YARN e HDFS.
Mentre la prima componente si occupa della gestione delle risorse del cluster e dell'assegnano di queste ad eventuali applicativi che ne fanno richiesta; HDFS è il filesystem distribuito predefinito di Hadoop, il quale permette lo sharding e la replicazione dei dati.

2.2 Apache Spark

Per quanto riguarda la parte di calcolo, è stato usato Spark, un motore di calcolo in-memory che dispone di librerie apposite per il machine learning su piattaforme distribuite. In questo caso sono state utilizzate delle librerie python per eseguire un algoritmo di regressione lineare per stimare il costo di un'automobile.

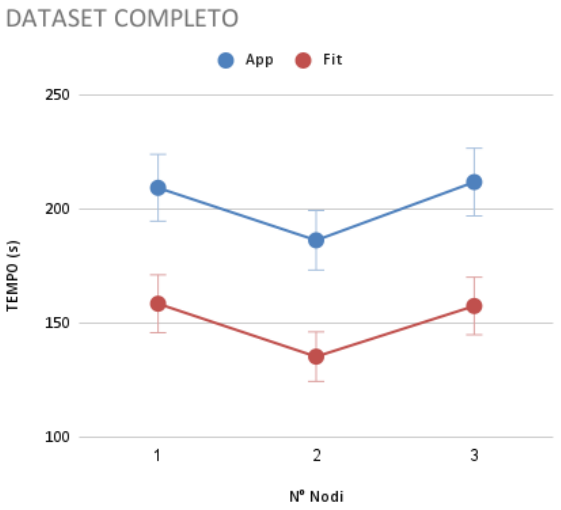
3. Caso di studio

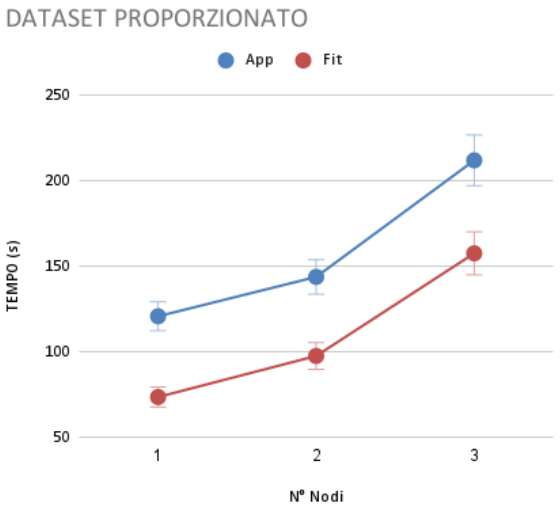
Nello svolgimento di questo progetto sono state analizzate due situazioni:

1. Analisi temporale al variare del numero di nodi del cluster (da 1 a 3) con dataset completo
2. Analisi temporale al variare del numero di nodi del cluster (da 1 a 3) con dataset proporzionato al numero di nodi attivi, nello specifico:
    - 3 Nodi con Dataset completo
    - 2 Nodi con 66% del dataset
    - 1 Nodo con 33% del dataset

In entrambi i casi è stato utilizzo lo stesso script per l'addestramento del modello dei dati, salvo per l'istruzione che svolgeva il campionamento del dataset.
Inoltre, sono state raccolte due metriche diverse: la prima riguardante il tempo di esecuzione registrato da YARN, mentre la seconda registrava solamente la fase di training dell'algoritmo.

3.1 Analisi dei dati





DATASET COMPLETO

Per quanto riguardo questo primo caso di studio, durante l'esecuzione dell'applicato è stato riscontrato una diminuzione significativa dei tempi, nel passare da un cluster a 1 nodo ad un cluster con 2 nodi; anche se il raddoppio della potenza di calcolo non ha portato ad una diminuzione dei tempi del 50%, il che è naturale visto l'aumento dell'overhead computazionale dovuto alla frammentazione del dataset e alla parallelizzazione delle operazioni, ha rispettato in parte i risultati attesi.
Nel passaggio da 2 a 3 nodi invece, la dimensione modesta del dataset non ha permesso di apprezzare un calo nei tempi di esecuzione; piuttosto i tempi legati alle operazioni necessarie alla distribuzione del carico computazionale sono diventati predominanti, aumentando i tempi di esecuzione ed uguagliandoli a quelli del cluster con un unico nodo.

DATASET PROPORZIONATO

In questo secondo caso di studio, i risultati reali si sono discostati ulteriormente da quelli attesi.
Le aspettative erano di osservare un tempo costante in tutte e tre le esecuzioni, al netto di un piccolo overhead all'aumentare dei nodi. Invece, anche in questo caso, le operazioni per il partizionamento dei dati e parallelizzazione hanno creato un overhead non trascurabile, facendo si che all'aumentare dei nodi aumentasse anche il tempo di esecuzione dell'applicazione.

Note

- Tutte le misurazioni riportate in figura sono la media di 5 misurazioni differenti per ognuna delle tre situazioni presenti nei due diversi casi di studio.
- Inoltre, in figura è anche riporta la deviazione standard delle misurazioni.
- Un ulteriore appunto riguarda le due metriche riportate in ogni figura: con il termine "App" si indica i tempi di esecuzioni ottenuti da YARN tramite chiamata alle API REST. Mentre con "fit" si indica il solo tempo di training dell'algoritmo, ottenuto tramite un file di log appositamente strutturato.
- Come si può notare dai grafici la differenza tra le due metriche rimane costante in tutti i casi studiati, questo è dovuto al fatto che tutte le operazioni precedenti all'istruzione di "fit()", compreso il campionamento, sono trasformazioni e vengono eseguite in modalità "lazy", ovvero quando la funzione di fitting prende in input il dataset trasformato.