Data science è la scienza che si occupa di estrarre conoscenza dai dati. Si basa sulle tecnologie informatiche e sulla statistica.

Il data scientist è un esperto di queste aree con una forte predisposizione verso il business e i dati. È capace di trasformare le informazioni nascoste nei dati in vantaggi competitivi per l’azienda. Il suo scopo finale è quello di creare nuovi modelli di business chiamati data drive economy.

Data science è un’evoluzione di Business Intelligence, ma lavora sia su dati strutturati che non strutturati ed è implementato su big data e su tecnologie NoSQL.

Data science è nata per bisogni del business, per la disponibilità di grandi dataset e per la presenza di tecniche di data analysis appropriate.

Nasce

* dalle pressanti esigenze degli utenti,
* dalla disponibilità di dati e
* dalla disponibilità di una grande potenza computazionale.

La technogym è un’azienda che aggiunge ai suoi macchinari dei sensori e un software, crea un ambiente social per fornire servizi aggiuntivi ai suoi clienti, inoltre mantiene il suo framework del software sul cloud.

Uno dei principali effetti dei big data è quello di far aumentare l’interesse per la data analysis. La maggior parte delle tecnologie relative ai big data sono nuove e le tecniche sono già conosciute, sono semplicemente state adattate alle nuove tecnologie.

I big data sono dataset con le seguenti caratteristiche:

* Volume: superano la capacità di processamento dei database
* Velocità: cellulari, dispositivi personali e transazioni IoT producono dati a una velocità maggiore che i sistemi informativi tradizionali
* Varietà: i dati sono estremamente eterogenei (testi da social network, dati provenienti da sensori, log di applicazioni). Possono essere strutturati (database relazionali), semistrutturati (documenti XML) o non strutturati (documenti di testo).
* Veracity (Accuratezza): la qualità dei dati è eterogenea e in molti casi

non è ben definito uno schema (scheme less) [enfatizza la non eterogeneità]

Con IoT si intende oggetti smart, che possono comunicare tra loro e accedere a informazioni aggregate da altri.

Data lake: un grande storage repository che mantiene i dati fino a che ce ne sarà bisogno, supporta analisi avanzate. Permette a data analisti di condurre ricerche e analisi.

È stato inizialmente concepito come archivio di dati, ma presto si è realizzato che senza processare e strumenti per la gestione l’utilità dei dati è limitata.

Il termine data lake spesso è sostituito con data platform o data ecosystem.

Oggi esistono cloud data platform per l’analisi e la trasformazione di dati. Questi tool evitano agli utenti di gestire la complessità tecnologica e quindi consente alle aziende a concentrarsi su aspetti funzionali.

I cloud sono un modello di trasporto per accessi su richiesta per condividere gruppi di risorse computazionali configurabili (come reti, db, software) che possono essere rapidamente assegnati e distribuiti con il minimo sforzo.

Il cloud computing è un modello computazionale che permette di accedere alle risorse hardware e software tramite richiesta e pagando per l’uso.

I servizi offerti dal cloud:

* Software as a service (SaaS): le applicazioni del provider degli utenti “runnano”, vengono eseguite, sulle infrastrutture cloud del provider. (es Gmail, Facebook)
* Platform as a service (PaaS): i clienti possono creare applicazioni personalizzate usando tool di programmazione supportati dal provider e distribuirle nell’infrastruttura cloud del provider (es Microsft Azure CosmosDB)
* Infrastructure as a service (IaaS): i clienti possono usare risorse computazionali all’interno dell’infrastruttura del provider (es Amazon EC2)

Le risorse sono fornite come un servizio sulla rete e può essere a 3 livelli diversi:

IAAS => infrastrutture as a service (hardware virtualizzato, server, storage, reti)

PAAS => platform as a service (esecuzione a runtime, database, web server, IDE)

SAAS => software as a service (app, mail app)

La forza dei cloud:

* scalabilità: la risorsa è disponibile quando il cliente ne ha bisogno, non ci sono delays durante l’espansione della capacità o spreco di capacità non usata
* non c’è bisogno di investire nel hardware: ogni cosa è già impostata e mantenuta dal cloud provider, il cliente risparmia tempo e costo.
* Paghi per quello che usi: se il servizio è richiesto per un periodo limitato, bisognerà pagare per solo quel periodo e le iscrizioni possono essere tolte in qualsiasi momento.
* Aggiornamenti e i disaster recovery (ripristino dei sistemi, a fronte di gravi emergenze che ne intacchino la regolare attività) sono automatizzati: Le distribuzioni delle risorse sono fatte automaticamente e gli aggiornamenti sono eseguiti senza costi
* Vi è flessibilità
* Aumenta la collaborazione
* Prevede aspetti di sicurezza

In un’architettura multicloud le applicazioni e i processi possono essere mantenuti su piattaforme cloud differenti. Si può ottenere vantaggi con una soluzione best of breed (vengono utilizzati i moduli di diversi vendor che meglio si prestano alle esigenze dell’azienda o che vengono considerati i migliori).

I rischi vengono attenuati: distribuendo sistemi critici su più cloud aumenta la tolleranza (fault tollerance)

Hadoop è un framework che permette a processi distribuiti su ampi dataset attraverso cluster di computer di usare semplici modelli di programmazione. Piuttosto che affidarsi sul hardware per il trasferimento ad alta disponibilità e affidabilità, la libreria è stata progettata per rilevare e gestire gli errori a livello di applicazione.

Architettura SMP

I tradizionali RDBMS (relational) sono basati su un’architettura simmetrica multi processing: diversi processori condividono la stessa RAM, lo stesso bus I/O, lo stesso disco.

Il limite dell’architettura dipende dal numero fisico di device che possono essere collegati e da collo di bottiglia sul bus.

Architettura MPP o shared nothing

In un’architettura massively parallel processing ci sono diversi processori, ognuno dotato della propria RAM e disco. Collaborano per risolvere un unico problema dividendosi i task indipendenti.

L’elaborazione dei dati avviene su nodi distribuiti, i quali processano i dati in parallelo e nell’ultimo nodo i dati prodotti vengono uniti insieme per ottenere l’output finale.

Il sistema è basato su un hardware ad alto livello e su software proprietario, mentre Hadoop è basato su hardware conveniente/economico e software libero. Hadoop usa codice imperativo, mentre MPP usa query dichiarative. L’SQL di MPP è più semplice e più produttivo e gli esperti di database costano meno di specialisti di Hadoop.

Architettura Grid Computing

È una collezione di risorse di computer situate in luoghi differenti che hanno un obiettivo in comune. Ogni nodo è impostato per eseguire diversi task/app. I computer sono più eterogenei e geograficamente dispersi rispetto ai cluster computers. Solitamente ogni architettura Grid computing è utilizzata per diversi scopi. Spesso hanno un general purpose grid middleware.

Architettura HPC

HPC si riferisce a un sistema massively parallel specificatamente devoto per risolvere task di cpu-intensive.

Le principali applicazioni sono simulazioni scientifiche, modellazione 3D, previsioni del meteo, …

In molti casi i processori condividono gli stessi dischi.

Scalabilità è l’abilità di un sistema, rete o processo di gestire la crescita di una quantità di lavoro o la sua abilità di aumentare per adattare la crescita.

La scalabilità può essere orizzontale o verticale. Scalare orizzontalmente (scale out) significa aggiungere più nodi al sistema, come l’aggiunta di un computer a un’applicazione distribuita. Scalare verticalmente (scale up) significa aggiungere risorse a un singolo nodo del sistema, solitamente aggiungendo cpu o memoria.

La scalabilità verticale è limitata dalla capacità dell’architettura a cui si devono aggiungere le risorse e dalla limitazione fisica delle risorse. Inoltre scalare orizzontalmente spesso è più conveniente.

Hadoop 1 (ad esempio map reduce) è nato per sostenere specifici task. È difficile da programmare, è robusto ed efficiente sulla batch ma non è adatto per i processi online.

Hadoop 2 (ad esempio yarn) è nato per rendere hadoop un framework general purpose. Le map reduce è solo uno dei possibili paradigmi utilizzabili, supporta SQL ed è adatto per processi online. Vi è una console di controllo grafica, connettori per software esterni.

I fattori d’impatto sulla scelta del cloud provider sono i costi, la completezza del cloud market place, la presenza di altri progetti ospitati.

L’analisi disincanto di Hadoop

Hadoop è un sistema distribuito, facilita l’amministrazione e la programmazione. Map reduce è un modello di programmazione che è facile da parallelizzare ma non tutti i problemi possono essere efficientemente parallelizzati con quell’architettura. Hadoop opera su cloud in quanto richiede molta potenza computazionale, molta memoria. Hadoop non è molto più efficiente delle architetture centralizzate (come Oracle DBMS), ma può essere scalato orizzontalmente a un prezzo minore.

Hadoop è utilizzato per creare modelli (data modelling), memorizzare e gestire dati. Hadoop impone un paradigma di programmazione con un limitato set di funzionalità.

Hadoop 1 vs Hadoop 2

Caratteristiche Hadoop 1: è ispirato da un progetto simile di Google basato su Map Reduce, è orientato alle esecuzioni in batch.

Caratteristiche Hadoop 2: è stato rilasciato nel 2013, risolve alcuni bottleneck e alcuni single point of failure. Può essere un data operating system aggiungendo Yarn (gestore delle risorse, scheduler delle applicazioni degli utenti). Hadoop 2 è inoltre aperto ad altri meccanismi computazionali diversi dalle Map reduce

Hadoop Distributed File System 1

È un file system distribuito progettato per essere eseguito su hardware economico. I suoi scopi sono individuazione dei fault, recovery automatico. È progettato più per un processo batch che per un’interazione con l’utente. Viene evidenziato l’alto throughput (frequenza con cui vengono trasmessi i dati) di accesso ai dati, piuttosto che una bassa latenza (tempo dall'input alla risposta). Le applicazioni che vengono eseguite su HDFS hanno molti data, in media un file ha dimensioni che varia dai gigabytes ai terabytes. Le applicazioni hanno bisogno di un accesso write once read many. Un file può essere creato, scritto e una volta chiuso non può essere modificato.

Hadoop Distributed File System 2

Mentre nella versione 1 tutti i dischi sono gestiti equamente in ogni nodo, la versione 2 sfrutta tutta la memoria perché è technology aware (a conoscenza). L’amministratore del sistema di cluster sarà in grado di configurare la memoria media disponibile su ogni nodo.

HDFS Federation

HDFS 2 implementa una federazione con lo spazio dei nomi (name spaces). Un name space è una gerarchia di file e cartelle. I metadata di ogni name space sono memorizzati nel nome del nodo. Il nome del nodo mantiene un mapping dei blocchi di dati ai data nodes. I name space usano gruppi di blocchi sotto un block pool. Un data node può fornire blocchi a più di un name space.

HDFS Federation caratteristiche:

Scalabilità: il nome del nodo tiene tutti i name space e locazioni di blocchi in memoria. La grandezza del Name Node heap limita il numero di file e il numero di blocchi indirizzabili.

Performance: Un nome di un nodo è un single point per memorizzare e gestire metadati. Può diventare un bottleneck per supportare un grande numero di file, in particolare un gran numero di piccoli file.

Disponibilità: puoi separare name space su diverse applicazioni

Manutenibilità, sicurezza e flessibilità: ogni name space è isolato e non è a conoscenza degli altri.

Cluster topology

I nodi sono organizzati in rack (armadietti) organizzati in data center.

Il contenuto dei file è diviso in grandi blocchi (default 128MB) e ogni blocco del file è indipendentemente replicato per migliorare le performance e la robustezza. In caso di non disponibilità, il sistema in modo trasparente ribilancia le repliche.

Un data block è la più piccola unità di dati indirizzabili nel nome del nodo. Localmente su un data node una richiesta di un data block sarà trasformata in diverse richieste OS blocks (pagine di disco). I blocchi OS che compongono un data block possono essere acceduti indipendentemente.

La località dei dati

Hadoop sfrutta la tipologia dei cluster delle repliche dei data block per applicare il principio di località dei dati: Quando le computazioni coinvolgono un grande set di dati è più economico e veloce spostare il codice dai dati piuttosto che i dati dal codice.

1 processo e dati sullo stesso nodo

2 processo e dati su diversi nodi ma nello stesso rack

3 processo e dati su diversi rack ma nello stesso data center

4 processo e dati in diversi rack e diversi data center

Un name space metadata include: l’albero dello spazio dei nomi e il mapping dei blocchi ai data node

Gli inodes registrano gli attributi come i permessi

Il record persistente dell’immagine memorizzata nel file system locale del nome del nodo è chiamata checkpoint. Il nome del nodo memorizza cambiamenti al HDFS in un log “write ahead” chiamato journal nel suo file system locale. Un nome del nodo secondario crea snapshot della cartella dell’informazioni del primo nodo. È usata per riavviare un primo nodo che fallisce.

High availability

In un cluster ad alta disponibilità, due separate macchine sono configurate come nome del nodo. Solo una è attiva mentre l’altra è in standby. Il nome del nodo in standby si mantiene informato sui metadata dello spazio dei nomi leggendo file di log scritto dal nome del nodo attivo o ricevendo le locazioni di data block.

Il meccanismo heart beat

Gli heart beat sono segnali periodici dei data node che vengono inviati al nome del nodo per farlo venire a conoscenza che loro sono attivi. L’assenza del heart beat triggera diverse azioni dentro al cluster. Se il nome del nodo non riceve l’heart beat dal data node esso è considerato inattivo e verrà creata una sua replica su diversi data nodi dei data block memorizzati in quel nodo.

YARN

È una piattaforma di gestione delle risorse responsabile della gestione delle risorse computazionali nei cluster e usando loro per schedulare le applicazioni degli utenti.

Un gestore delle risorse globale che le gestisce attraverso tutte le applicazioni del sistema. Le risorse sono assegnate e usate a un container il quale ha memoria, cpu, disco, rete.

Il nodo manager è uno slave node il quale è responsabile di contenere, monitorare l’uso delle risorse e riportare lo stesso al resource manager.

Un application manager ha compito di negoziare le risorse da un resource manager e lavora con il nodo manager a eseguire e monitorare i task.

Il resource manager ha due componenti: lo scheduler e l’application manager. Lo scheduler è responsabile per allocare risorse in applicazioni in esecuzione. Lo scheduler non monitora o tiene traccia dello stato delle applicazioni, offre una garanzia di riavviare task falliti dovuti a errori delle applicazioni o del hardware.

L’application manager è responsabile di accettare le richieste dei job, negoziare il primo container per l’esecuzione di applicazioni specifiche e fornisce un servizio di riavvio dell’application manager container in caso di errore.

Application Master

Permette a Yarn di essere più scalabile (molte funzionalià sono distribuite su diversi application managers, quindi il resource manager non è più un bottleneck nel cluster) e più aperto (può supportare molti frame

work). L’allocazione e deallocazione dei container può avvenire in maniera dinamica man mano che l’applicazione viene eseguita.

YARN WALKTRHOUGH

L’esecuzione dell’applicazione consiste nei seguenti step:

Invio dell’app, l’avvio dell’istanza dell’application manager che gestirà l’app.

1 un programma di un cliente invia l’app

2 il resource manager negozia uno specifico container nel quale avviare l’application manager e lanciare l’app.

3 l’application manager registra con il resource manager

4 durante le normali operazioni l’application manager negozia un appropriato container delle risorse tramite il protocollo di richiesta della risorsa.

5 se l’allocazione del container va a buon fine, l’application manager lancia il container.

6 il codice dell’app viene eseguito dentro il container e poi vengono fornite le informazioni necessarie all’application manager (progressi, stato)

7 durante l’exe dell’app, il cliente può inviare il programma che comunica direttamente con l’application manager per ottenere lo stato, i progressi

8 una volta che l’applicazione ha terminato l’esecuzione, l’application manager si disiscrive con il resource manager e si spegne, permettendo al proprio container di essere riutilizzato