Università di Pisa

Corso di Laurea in Informatica Umanistica

Didactic Data Mining

Un'applicazione web per E-Learning e per la risoluzione di esercizi

Candidato: Maurizio Ricci

Relatori: Dr. Anna Monreale, Dr. Riccardo Guidotti

Correlatore: Dr. Enrica Salvatori

Anno Accademico 2016-2017

Indice

1 Introduzione

L'obiettivo di questa tesi è quello di descrivere in maniera dettagliata il motivo e il modo in cui ho realizzato Didactic Data Mining (DDM) ovvero un'applicazione web il cui scopo è quello di risolvere determinati esercizi di Data Mining per conto dell'utente in modo tale da favorire maggiormente l'apprendimento degli algoritmi studiati a lezione.
L'idea di questo progetto nasce con lo scopo di aiutare gli studenti del corso di Laurea Magistrale in Informatica e Informatica Umanistica per quanto riguarda l'apprendimento della materia Data Mining. Il progetto si basa su una libreria software esistente, volta alla risoluzione di determinati tipi di esercizi della suddetta materia. Tale libreria offre quindi all'utente gli strumenti per poter risolvere gli esercizi all'interno di un interprete Python. Tuttavia, proprio per la sua natura stessa di libreria software necessita di essere utilizzata all'interno di un programma scritto in Python; si presentano quindi due inconvenienti. Il primo problema riguarda la conoscenza e il rispetto della sintassi dei comandi da impartire per poter eseguire i vari task. La seconda difficoltà risiede nel dover avere a disposizione un elaboratore con Python installato e configurato correttamente, tutto ciò limita molto l'utilizzo della libraria software da parte degli utenti. Da questi due problemi principali scaturisce l'idea di creare un'applicazione web per poter sottomettere i task al risolutore di esercizi, senza però avere più gli inconvenienti detti in precedenza poiché viene implementata un'interfaccia grafica e essendo un sito web per sua natura risulta essere usabile da qualsiasi dispositivo computer o telefono che sia.

Lo scopo dell'applicazione Web è quello di fornire all'utente un'interfaccia grafica, ovvero un punto d'incontro tra utente e software, per l'interazione con la libreria che si occupa di risolvere gli esercizi cercando di essere il più possibile *user friendly*.
A partire da questa libreria software è stato creato un *web server* in Python il quale si occupa di ricevere i dati ei comandi forniti dall'utente, di processarli e di restituire una risposta testuale e grafica all'utente mediante l'uso del risolutore di esercizi. Tramite l'utilizzo di Javascript le richieste vengono inviate al server pronte per essere elaborate e le risposte provenienti del server vengono presentate all'utente sotto forma di testo e/o immagini.

Durante lo svolgimento di tale progetto mi è stato possibile acquisire maggiori capacità nello sviluppo di siti web basati su framework, in particolare l'apprendimento dell'uso di Flask, un web server di nuova generazione basato sul paradigma MVC (vedi Sezione 4.6.1). Sono riuscito poi a acquisire sempre maggiore conoscenza delle funzionalità della libreria jQuery.

La struttura dei contenuti che verranno presentati è la seguente: verrà prima presentato il capitolo inerente Data Mininig in cui si fornirà un'introduzione alla suddetta materia e agli algoritmi utilizzati nel progetto. Sarà poi la volta di andare a fornire una panoramica generale del concetto di E-learning, il quale risulta essere strettamente connesso all'applicazione sviluppata. Mentre l'infrastruttura e le tecnologie utilizzate per sviluppare l'applicazione sono trattate a parte cosi come la progettazione del sito e le soluzioni grafiche adottate nel progetto.

2 Data Mining

2.1 KDD Process

Con Data Mining si intende l'insieme di tecniche e metodologie che hanno per oggetto l'estrazione di un sapere o di una conoscenza a partire da grandi quantità di dati (attraverso metodi automatici o semi-automatici) e l'utilizzo scientifico, industriale o operativo di questo sapere[1].

L'estrazione di conoscenza da grandi quantità di dati è un'operazione che spesso si ottiene andando a recuperare i dati dai database dove i dati sono rappresentati in modi differenti rispetto a quelli desiderati. Occorre quindi una fase preliminare prima di poter estrarre informazioni dai dati grezzi, dopodiché è possibile iniziare a estrarre conoscenza dai dati. L'intero processo è chiamato KDD Process (*Knowledge Discovery process*), questo metodo si rivela particolarmente interessante per ricercatori e data scientist che si occupano di machine learning, intelligenza artificiale, data visualization, statistica.

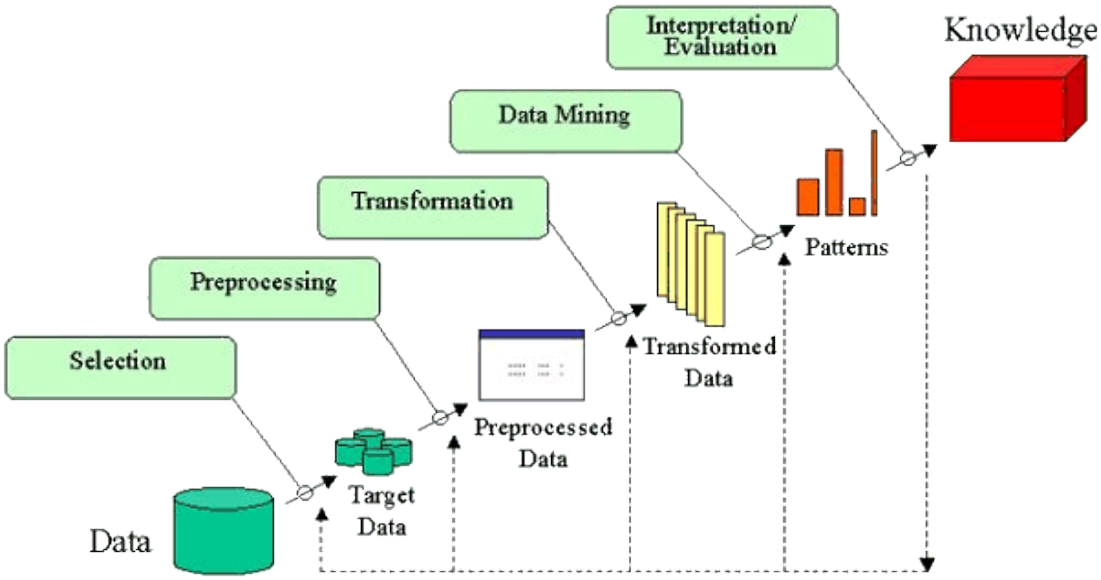


Figura 1. KDD Process illustrazione fasi

Tale processo è articolato in varie fasi:

1. Sviluppo e analisi dei requisiti dell'applicazione ponendo l'attenzione sullo scopo finale del progetto.
2. Creazione del modello di dati interessanti da utilizzare e estrazione dei dati dal database. In questo passaggio solitamente si fa uso di interrogazioni al da-

[1]https://it.wikipedia.org/wiki/Data_mining

tabase per ottenere i dati richiesti in modo da raggrupparli o organizzarli tra loro.

3. Pulizia dei dati. Rimozione di elementi non interessanti per le analisi e gestione di possibili dati incompleti o mancanti.
4. Riduzione del numero di variabili da tenere sotto controllo mediante opportune assunzioni.
5. Scelta del *task* da portare a termine. Se si tratta quindi di clustering, classificazione di oggetti o di estrazione di schemi ricorrenti.
6. Scelta dell'algoritmo di data mining opportuno e configurazione delle opzioni di tale algoritmo.
7. Esecuzione dell'algoritmo con i dati raffinati nei punti precedenti.
8. Visualizzazione e interpretazione dei risultati calcolati dall'algoritmo.
9. Acquisizione di conoscenza. I risultati ottenuti vengono documentati sotto forma di relazione in modo da poter essere facilmente accessibili. Inoltre, in questa fase si procede anche alla risoluzione di possibili conflitti con risultati di altre analisi passate.

2.2 Tecniche di Data Mining

2.2.1 Clustering

Un *cluster* è una classe o un insieme di oggetti che condividono varie caratteristiche comuni. L'intento globale di un task di clustering è quello di replicare il normale comportamento del cervello: esso è bravo nel suddividere oggetti in gruppi (*cluster*) e nell'assegnare altri oggetti a questi gruppi in base a determinate proprietà.
La suddivisione in *cluster (clustering)* si può applicare in vari settori

- **Biologia:**

  I ricercatori hanno speso un sacco di anni nel creare delle tassonomie per classificare ogni forma di vita. Questo lavoro potrebbe essere svolto mediante l'analisi dei *cluster*. Il *cluster* potrebbe essere inoltre usato per raggruppare i geni studiati in base a certe caratteristiche.

- **Information retriveal:**

  Il web si compone di miliardi di pagine web, occorre quindi un modo per riuscire a districarsi tra l'elenco dei risultati per ottenere ciò che si cerca. Il clustering può essere usato per raggruppare i risultati di ricer-

ca in gruppi, ciascuno di essi approfondisce una determinata parte della query (chiave di ricerca)

- **Business:**

  I clienti potrebbero essere suddivisi automaticamente in sottogruppi in modo tale da poter operare analisi finanziarie più dettagliate e mirate.

L'analisi dei *cluster* raggruppano i dati degli oggetti facenti parte dei gruppi solamente sulla base delle informazioni trovate nei dati che descrivono le entità e le loro relazioni. L'obiettivo è quello di raggruppare tra loro gli oggetti più simili in modo tale che un oggetto facente parte di un gruppo sia il più possibile diverso da altri oggetti appartenenti a altri gruppi o *cluster*. Non è però nota una precisa definizione di *cluster*, si possono fare infatti più tipi di raggruppamenti in base al livello di dettaglio desiderato.
Principalmente esistono due tipi di clustering *Hierarchical* e *Partitional*.
Un *cluster* di tipo *partitional* è un *cluster* diviso in partizioni tra loro non sovrapponibili. Se lasciamo che questi *cluster* possano essere divisi in *sub-cluster* organizzati secondo una struttura a albero (gerarchica) il risultato sarà un *cluster* di tipo *Hierarchical* (gerarchico). La seguente figura mostra le differenze tra i due tipi di clustering.

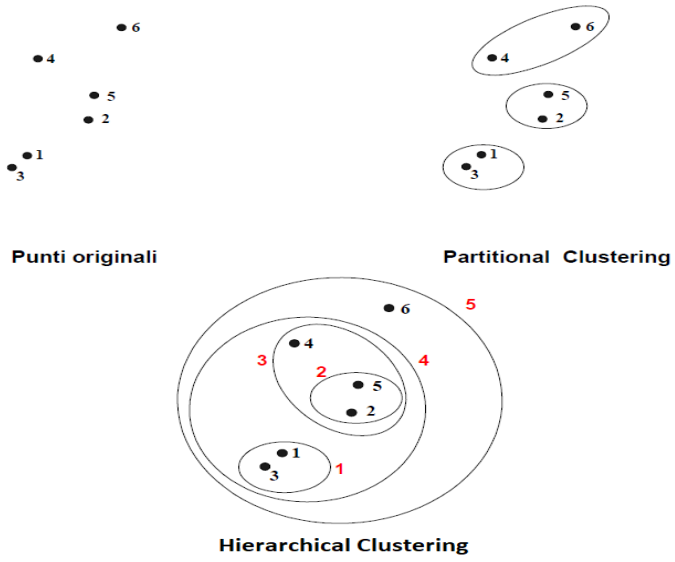


Figura 2. Differenti modalità di Clustering

2.2.2 Differenti tipi di *clustering*

Esistono diversi algoritmi di clustering che si possono dividere in 3 categorie:

- **Ben separati:**

  Un *cluster* è costituito da un insieme di punti tali che per ogni punto appartenente ad un *cluster*, questo punto è più vicino (o più simile) ad ogni altro punto del proprio *cluster* rispetto a tutti gli altri punti

- **Basati su un prototipo:**

  Un *cluster* è un set di oggetti nel quale ogni oggetto è più simile al prototipo (o punto rappresentativo) che definisce il *cluster* corrente piuttosto che al prototipo definito in altri *cluster*. Spesso questo prototipo è un centroide (detto anche baricentro), in geometria rappresenta la posizione media di tutti i punti di una figura.

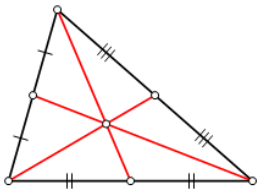


Figura 3. Centroide o baricentro in un triangolo

- **Basati sulla densità:**

  Un *cluster* viene definito come una regione densa di un certo tipo di oggetti, questa regione deve poi essere circondata da altre regioni a bassa densità

2.2.3 Classificazione

Nella classificazione l'obiettivo da realizzare è quello di fare delle predizioni basandosi su dei dati già osservati (*training dataset*). Similmente alle persone si usa cosi l'esperienza pregressa per poter formulare delle ipotesi. Ciò può essere fatto andando a cercare i giusti tratti caratteristici di ogni possibile oggetto osservabile. È possibile quindi andare a fornire dei dati in cui vengono elencate le *feature* specifiche per poter affermare che un oggetto sia di un certo tipo.

Per esempio, un tratto distintivo per poter classificare una persona in maschio o femmina potrebbe essere data dalla lunghezza dei capelli. Fissare a 15 cm la soglia oltre la quale persone con i capelli più lunghi saranno classificate come femmine, può essere un primo approccio ma non basta.

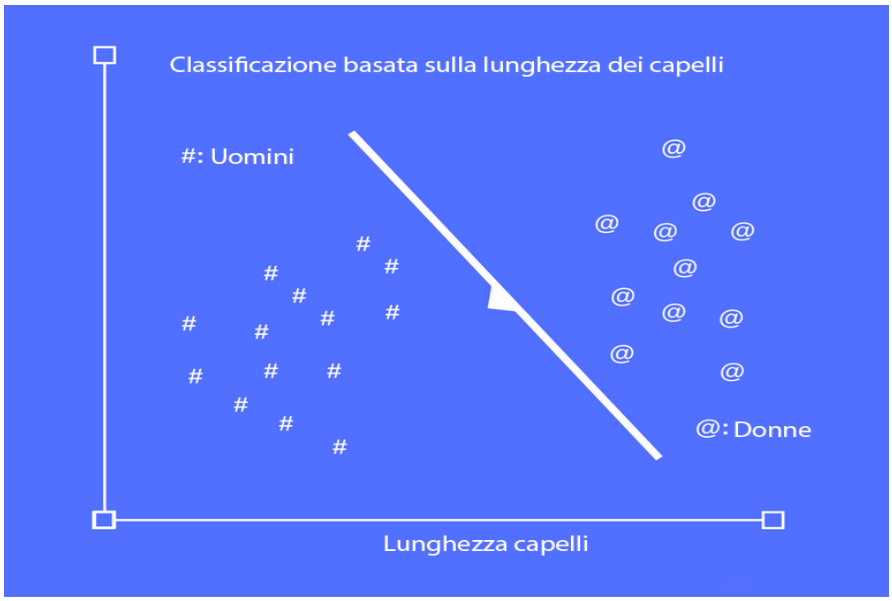


Figura 4. Classificazione basata su lunghezza dei capelli

Più *features* il sistema ha a disposizione più è in grado di effettuare previsioni migliori, non tutte infatti sono decisive per la scelta finale, è il sistema che si occupa anche di stabilire un peso o un grado di rilevanza per ogni caratteristica osservata.
Una volta passati i dati con le loro *features* il sistema in una prima fase (fase di *training*) osserva come queste *features* influenzino il risultato in modo tale da creare un suo modello da utilizzare per classificare nuovi oggetti in base alle caratteristiche che presentano, usa ciò l'esperienza pregressa per formulare ipotesi.

I sistemi di classificazione automatica trovano applicativi in vari campi tra cui:

- Identificazione dello spam nelle e-mail
- Identificazione di celle tumorali
- Sentiment analysis
- Applicazioni per il riconoscimento del volto
- Individuazione della presenza di pedoni sulla strada

Sebbene il clustering possa apparire simile alla classificazione c'è da notare che quest'ultima non rappresenta gli oggetti scandendoli per similarità ma cercando di trovare quante più regole generali possibili per giungere al risultato.

2.2.4 Pattern mining e regole associative

Pattern mining consiste nell'uso o sviluppo di algoritmi di data mining specifici per la scoperta di eventuali *pattern* o regolarità nei dati che si presentano periodicamente, e che quindi potrebbe essere interessante prenderli in considerazione al momento di dover prendere decisioni importanti.
Per esempio, queste analisi possono rivelarsi utili all'interno delle transazioni effettuate nei negozi. Analizzando il comportamento dei clienti, può essere possibile scoprire che alcuni tipi di acquirenti comprano determinate cose ogni fine settimana, cose come formaggio o vino. Tutto ciò può essere usato per promuovere certi tipi di prodotti o può servire per prendere certe decisioni di mercato.

Una volta estratti dai dati, i *pattern* permettono la creazione di regole associative. Regole dalla forma $X \rightarrow Y$, ciò se si osserva X allora si presenterà anche Y, dove X e Y possono essere singoli oggetti oppure transazioni (insiemi di oggetti). Di nuovo questa conoscenza in settori finanziari o di mercato, uno scenario tipico può essere sempre quello di un negozio, in cui si vanno a estrarre delle relazioni tra i prodotti venduti. È possibile magari dire che chi acquista un certo prodotto molto spesso ne compra anche un altro, risulta quindi utile per possibili promozioni o sconti sui prodotti. Uno degli algoritmi più popolari per risolvere questo tipo di problema è Apriori, il quale riesce a risolvere il problema della complessità computazionale che si viene a creare nel momento in cui bisogna analizzare tutte le possibili accoppiate di transazioni.

2.3 Algoritmi usati nel progetto

2.3.1 Kmeans

È un algoritmo di clustering partizionale basato su un prototipo, in questo caso il prototipo è rappresentato da dei centroidi (Figura 3. Centroide o baricentro in un triangolo).
All'inizio l'algoritmo richiede un parametro K, il numero di centroidi da utilizzare, per ognuno bisogna specificare anche le sue coordinate X, Y sul piano cartesiano. Successivamente i punti del *dataset* (o collezione di dati) vengono assegnati al centroide più vicino andando a formare un *cluster*. Poi il centroide di ogni *cluster* viene aggiornato sulla base dei punti facenti parte del *cluster* (calcola il punto medio). Vengono infine ripetuti i passi di assegnamento dei punti ai centroidi e di aggiorna-

mento di questi ultimi. L'algoritmo conclude la sua esecuzione quando il nuovo assegnamento dei punti ai centroidi non produce cambiamenti rispetto all'assegnamento precedente. C'è da precisare che nell'implementazione fornitami di K-Means è possibile specificare solo due centroidi ovvero K=2.

Nella figura sottostante (Figura 5. Iterazioni Kmeans) è illustrato il funzionamento dell'algoritmo: all'inizio divide i punti in due *cluster* arbitrari, dove per ogni insieme calcola il suo punto medio. Successivamente i punti vengono riassegnati in base al gruppo del centroide più vicino. All'iterazione seguente vengono nuovamente ricalcolati i centroidi, se qualcosa è cambiato si replica il processo di assegnamento dei punti, altrimenti la computazione è terminata.

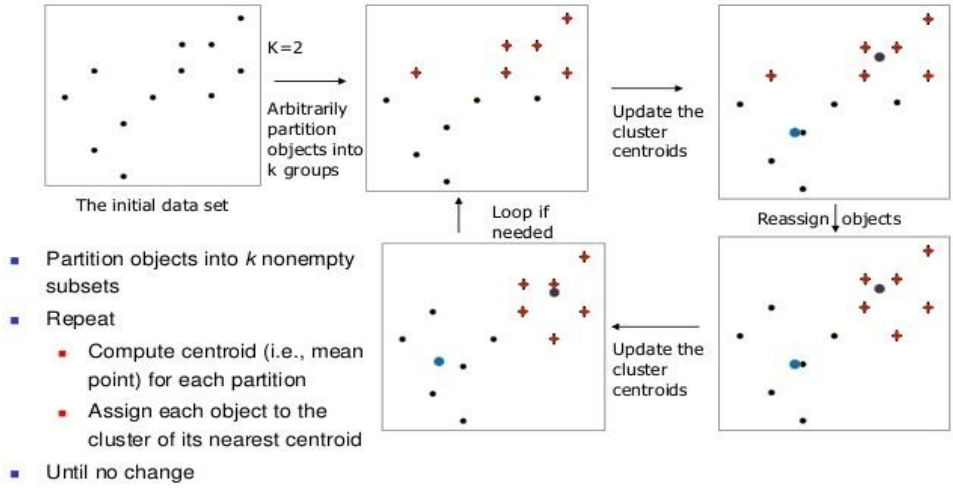


Figura 5. Iterazioni Kmeans

2.3.2 Dbscan

Questo algoritmo è basato sulla densità, ovvero va alla ricerca di regioni di punti ad alta densità separate da regioni di bassa densità. È necessario però trovare un modo per stimare la densità dell'area a cui appartiene un punto detto P; in genere viene utilizzato un approccio *center based*, ovvero la misura della densità si ottiene contando i punti iscritti in una circonferenza di raggio EPS e di centro P. Il raggio viene specificato dall'utente. I punti identificati da questo algoritmo sono divisi in tre gruppi:

1. ***Core points*:** vengono chiamati cosi i punti interni a una zona a alta densità. Questi punti devono inoltre superare un certo numero di punti imposto dal parametro (MinPts)

2. **Border points:** rappresentano quei punti interni a una zona ad alta densità ma che non sono abbastanza per costituire un *core point*, poiché non sono presenti almeno MinPts punti nel suddetto *cluster*
3. **Noise points:** sono tutti quei punti che non sono né *border points* né *core points*

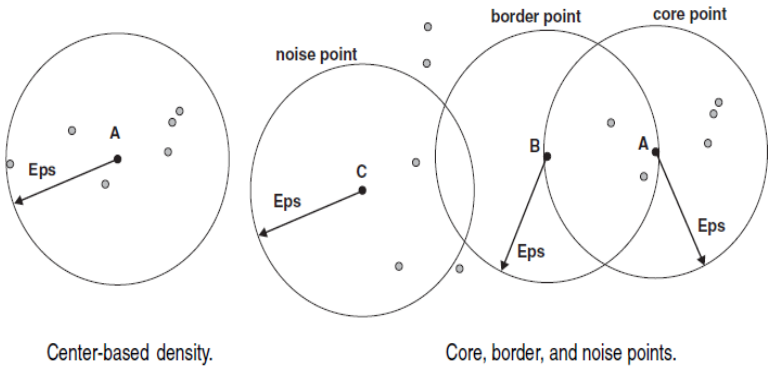


Figura 6. Noise-border-core points

L'algoritmo all'inizio va a distinguere i punti trovati in base alle 3 categorie dette sopra (*Core*, *Noise*, *Border points*), i *noise points* vengono poi scartati. Successivamente vengono collegati tra loro tutti i *core points* che non distano più di EPS tra loro (dove EPS è un parametro dell'algoritmo), in questo modo i punti che sono stati collegati diventano *cluster*. Infine, si assegnano i *border points* al *cluster* più vicino a loro.

2.3.3 Decision Tree

Lo scopo di questo algoritmo è quello di riuscire a fare delle previsioni in base a un certo numero di dati fornitogli. L'algoritmo presuppone un insieme finito di classi usate per classificare gli oggetti che gli verranno passati. Decision Tree o Classification Tree sono rappresentati per mezzo di un albero in cui ogni nodo interno è segnato per mezzo di una certa *feature* (caratteristica). Gli archi che collegano i nodi sono marcati con uno dei possibili valori delle caratteristiche. Ogni foglia (nodo finale

senza altri archi) dell'albero è rappresentato da una classe, viene anche indicata la probabilità stimata della previsione.
Per far sì che il sistema funzioni e sia in grado di effettuare previsioni, occorre prima fornire un modello di addestramento, viene passato un modello contenente vari casi di test in cui sono presenti tutte le caratteristiche. L'algoritmo andrà quindi a studiare come le singole caratteristiche influenzino il meccanismo di classificazione e a andrà a estrarre un peso per ogni feature; più è alto questo peso e più sarà affidabile la scelta. Ad esempio, in un sistema che si occupa di classificare nuove specie in mammiferi o non mammiferi, la temperatura corporea risulta essere una caratteristica molto importante per la scelta finale, potrebbe portare subito a una decisone finale (vedi Figura 7. Esempio Decision Tree - Classificazione specie).

Dopo aver analizzato il modello di addestramento l'algoritmo è pronto per fare previsioni basate su scenari già osservati o anche su nuovi. Più il modello di addestramento è completo e generale più saranno accurate le previsioni.

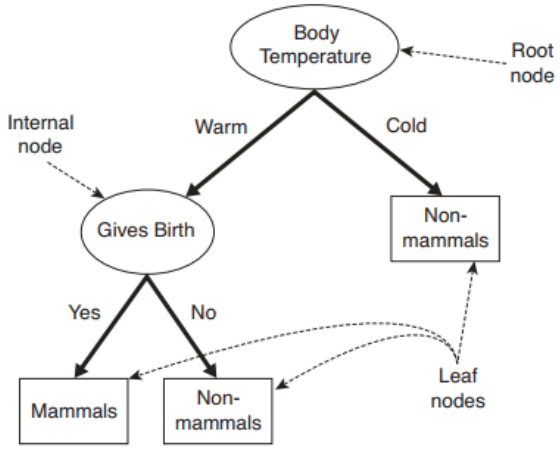


Figura 7. Esempio Decision Tree - Classificazione specie

2.3.4 Hierarchical

Esso è un algoritmo di clustering di tipo gerarchico (vedi Sezione 2.2). Esistono due modalità di esecuzione:

1. **Agglomerative:** All'inizio tutti i punti sono visti come tanti *cluster*, ad ogni passo i *cluster* più vicini vengono fusi insieme. Serve quindi un modo per calcolare la prossimità tra *cluster*.
2. **Divisive:** Tutti i punti fanno parte di un grande *cluster*, ad ogni passo il *cluster* viene diviso in sempre più parti. Questa modalità non è trattata all'interno della tesi.

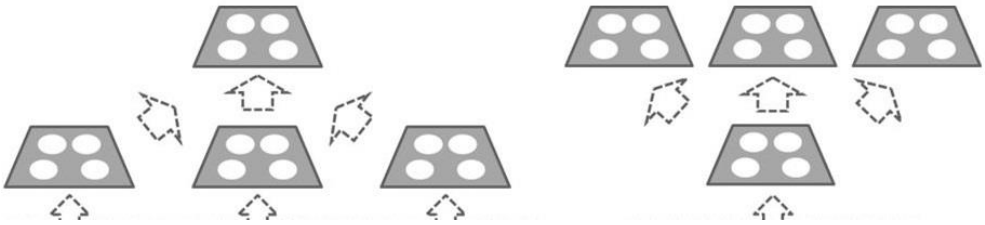


Figura 8. Hierachical agglomerative e divisive a confronto

La Figura 8. Hierachical agglomerative e divisive a confronto illustra la differenza tra *agglomerative* e *divisive*: nel primo caso i punti sono visti come tanti *cluster* da fondere via via in uno più grande, nell'altro caso accade il contrario ovvero un *cluster* viene diviso in tanti piccoli gruppi.

Esistono principalmente tre modalità per calcolare la prossimità tra *cluster*:

1. **Min:** la prossimità tra due cluster è data dalla distanza euclidea tra i due punti più vicini, ciascuno preso da un *cluster* differente
2. **Max:** calcola la distanza euclidea tra i due punti più lontani, ciascuno preso da un *cluster* differente
3. **Group average:** per ogni coppia di punti viene calcolata la distanza euclidea. Alla fine, viene fatta la media delle distanze calcolate tra tutti i punti dei due *cluster* in esame.

Viene poi proposta la rappresentazione grafica dei 3 metodi (Figura 9. Misure di prossimità tra cluster: Min, Max, Avg).

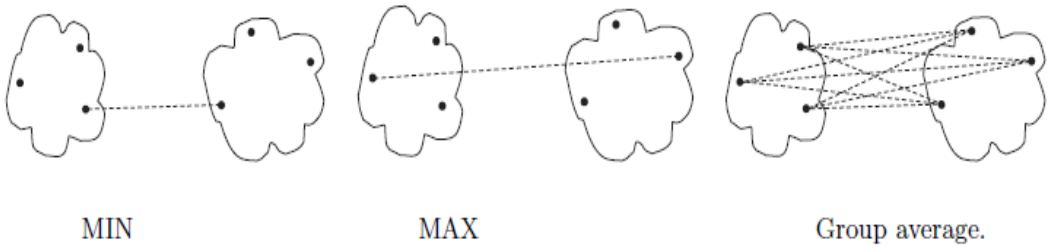


Figura 9. Misure di prossimità tra cluster: Min, Max, Avg

Una volta determinate le modalità per il calcolo della prossimità tra *cluster* è possibile iniziare il *processing* dei dati. Quindi all'inizio sono presenti tanti piccoli gruppi, per ciascuna coppia viene calcolata la distanza secondo le modalità appena definite e vengono fusi insieme i due *cluster* più vicini andando a formare un gruppo più grande. Viene ripetuto cosi il procedimento finché non sono stati uniti tutti i *cluster*. La figura sottostante (Figura 10. Schema Hierarchical) rappresenta il concetto appena descritto; assume che all'inizio le coppie B-C e D-E siano quelle più vicine tra loro. In ordine abbiamo poi che DE risulta essere vicino a F, realizzando un'altra fusione DEF, la quale poi viene unita alla precedente BC, come ultimo *cluster* viene aggiunto che si suppone essere il più lontano.

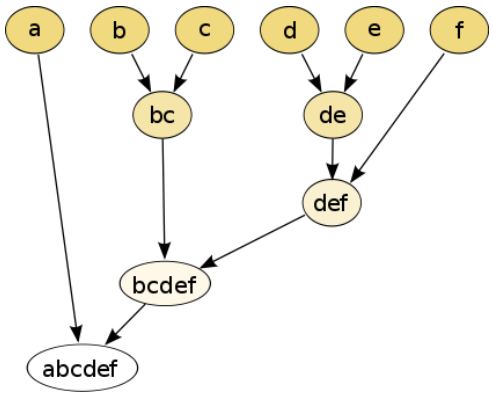


Figura 10. Schema Hierarchical

2.3.5 Apriori

L'algoritmo Apriori è utilizzato per la ricerca di associazioni frequenti di oggetti. Dati vari insiemi di oggetti (*item set*) costituito dai nomi degli oggetti, Apriori è in grado di riuscire a calcolare quali saranno le associazioni più frequenti dei suddetti oggetti.

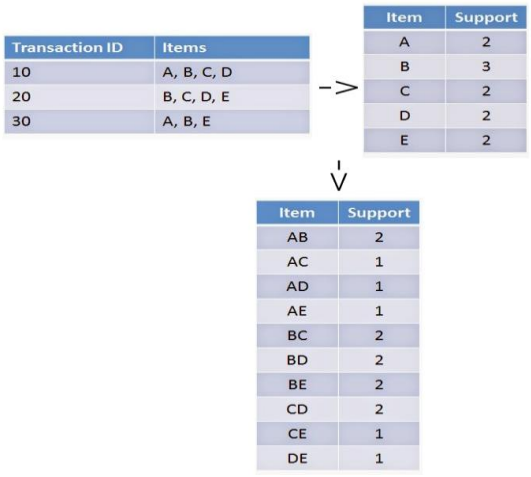
Le associazioni frequenti vengono calcolate tenendo a mente un semplice teorema "Se un oggetto è frequente allora i sottoinsiemi di quel dato oggetto saranno frequenti anche loro", da qui deriva il nome Apriori, è possibile conoscere appunto a priori se un sotto insieme di un certo oggetto sarà interessante per il risultato oppure no. Questo accorgimento è usato al passo 1 durante la fase di *prune* per riuscire a ridurre in modo significativo il costo computazionale del calcolo di tutte le possibili accoppiate degli oggetti.

All'inizio l'utente fornisce un parametro *minimum support* che specifica la frequenza minima di un oggetto per essere preso in considerazione. Poi viene fornito l'insieme degli oggetti su cui operare (*item set* o transizioni) (vedi Figura 12. Apriori passo 1 - creazione gruppi – tabella 1). Adesso Apriori può iniziare a calcolare il risultato, per farlo si divide in due passi:

- **Passo 1:** si combinano tra loro tutti gli oggetti distinti presenti negli *item set* (vedi Figura 12. Apriori passo 1 - creazione gruppi). Occorre poi raggrupparli e contare quante volte gli oggetti preseti in quel dato gruppo si presentano nelle transizioni iniziali fornite dall'utente; se il numero di volte che un gruppo si presenta non supera un certo valore imposto dall'utente (*minimum support*) allora si scarta l'accoppiata appena calcolata (fase di *prune*) (vedi Figura 11. Apriori passo 1 - rimozione elementi. Si ipotizza minimum support ≥ 2). Si ripete nuovamente il passo 1 usando come *input* le accoppiate rimanenti, se non rimane nessuna accoppiata o se non si possono combinare nuovamente gli elementi si passa alla fase successiva.
- **Passo 2:** Adesso è il momento della generazione delle regole, ovvero sulla base dei dati osservati il sistema può osservare le varie implicazioni che gli oggetti hanno tra loro. Per calcolare le regole vengono usati come dati quelli estratti al punto 1, studiando quindi le accoppiate frequenti dei vari oggetti è possibile stabilire le correlazioni tra gli oggetti (regole). Una regola si esprime sotto la seguente forma: $X \Rightarrow Y$.

  Significa che se X è presente allora anche Y sarà presente con una certa probabilità. Questo dato si rivela molto utile durante le indagini di mercato, in un

ipotetico caso si può inferire per esempio che un cliente che acquista cereali, acquista anche latte con probabilità 0.68.



Transaction ID	Items
10	A, B, C, D
20	B, C, D, E
30	A, B, E

Item	Support
A	2
B	3
C	2
D	2
E	2

Item	Support
AB	2
AC	1
AD	1
AE	1
BC	2
BD	2
BE	2
CD	2
CE	1
DE	1

Figura 12. Apriori passo 1 - creazione gruppi

Item	Support
AB	2
BC	2
BD	2
BE	2
CD	2

Figura 11. Apriori passo 1 - rimozione elementi. Si ipotizza minimum support ≥ 2

3 E-Learning

Il sito web realizzato si inserisce nella categoria di *E-Learning* in quanto lo scopo di tale sito è quello di aiutare gli studenti nell'apprendimento del corso di Data Mining. Infatti con il termine inglese *E-learning* s'intende l'uso delle tecnologie multimediali e di Internet per migliorare la qualità dell'apprendimento facilitando l'accesso alle risorse e ai servizi, così come anche agli scambi in remoto e alla collaborazione a distanza[2].

L'apprendimento online è un processo di formazione che implica l'utilizzo delle tecnologie di rete per favorire l'apprendimento. In quest'ottica gli elementi principali nella progettazione di contenuti erogabili via rete, i quali rendono la formazione a distanza diversa dai tradizionali corsi sono tre:

- **Interattività:** vale a dire la necessità di coinvolgere lo studente, generalmente avvalendosi del *learning by doing* ovvero imparare attraverso il fare.
- **Dinamicità:** ovvero il bisogno da parte del discente di acquisire nuove competenze mirate *just in time.*
- **Modularità:** ossia la possibilità di organizzare i contenuti di un corso secondo gli obiettivi formativi e le necessità dell'utenza.

Nel progetto di tesi quindi tutte e 3 queste caratteristiche vengono rispettate a pieno: L'interattività consiste nel provare a risolvere gli esercizi a mano e poi osservare la soluzione corretta, fornita in tutti i suoi passaggi; oppure gli utenti possono apprendere al meglio iniziando a fare esperimenti sui parametri degli algoritmi andando scoprire come cambia il risultato al variare dei dati o delle opzioni fornite in input. Mentre dinamicità e modularità sono possibili mediante una semplice e intuitiva selezione dell'algoritmo da utilizzare e da apprendere. Inoltre, ogni pagina ha un *link* diretto alle appropriate voci di Wikipedia in cui viene descritto in dettaglio il funzionamento di tale algoritmo in modo tale da poter offrire un rapido supporto allo studente nel caso avesse necessità di chiarire qualche suo dubbio nato sul momento.

[2]definizione dal documento COM (2001)172 della Commissione delle Comunità Europee del 28 marzo 2001, http://ec.europa.eu/transparency/regdoc/rep/1/2001/IT/1-2001-172-IT-F1-1.Pdf

4 Infrastruttura e tecnologie utilizzate

All'interno dell'applicazione web le pagine che la compongono sono state realizzate in HTML5 mentre la grafica è stata definita utilizzando i fogli di stile CSS.

4.1 HTML

HTML5 (HyperText Markup Language) è un linguaggio di markup utilizzato per trasformare documenti testuali in pagine web. Con il termine markup, originario dell'ambiente tipografico, si allude ad un insieme di regole che descrivono i meccanismi di rappresentazione, strutturali, semantici di un testo.
Esistono due tipologie di markup:

- **procedurale*:*** specificano quali sono le procedure di trattamento del testo e indicano le istruzioni da eseguire affinché la porzione di testo referenziata possa essere visualizzata; in questo tipo di markup è possibile che alcuni tag incorporino già delle istruzioni inerenti allo stile di un elemento.
- **descrittivo:** lascia che sia il software a scegliere quale rappresentazione debba essere applicata al testo. Il loro compito è solo quello di descrivere che tipo di elemento è stato marcato (paragrafi, titoli, citazioni…)

HTML è un linguaggio di markup descrittivo, e una tra le più importanti caratteristiche di quest'ultimo è che assicura che tra la struttura e la visualizzazione del testo vi sia una corretta separazione, in quanto è buona norma applicare lo stile mediante la creazione di un foglio di stile CSS (vedi Sezione 4.2) da incorporare poi nella pagina HTML.
L'HTML, come precedentemente detto, viene impiegato per formattare e per impaginare i documenti ipertestuali. Esso è stato sviluppato da Tim Berners-Lee alla fine degli anni '80. Questo linguaggio di formattazione descrive le modalità di impaginazione base di una pagina web, serve a creare le basi del *layout* di un sito web. Gli elementi che si possono creare sono quindi quelli base e comuni quali paragrafi di testo, citazioni, bottoni, riquadri per immagini, titoli, sottotitoli, form.
Nonostante quanto si possa pensare dalla definizione di markup procedurale bisogna precisare che HTML non fa parte della categoria dei linguaggi di programmazione. HTML consente di presentare all'utente immagini, video, testo come potrebbe accadere in un normale programma dotato di interfaccia grafica; la grande differenza ri-

siede nel fatto che HTML è privo di strutture di controllo tipiche dei linguaggi di programmazioni quali istruzioni condizionali (*IF-ELSE*), cicli (*FOR, WHILE*).

Una pagina web realizzata con HTML può presentare diverse strutture, ma quella basilare è costituita da un'intestazione e dal corpo del documento; ogni documento HTML inizia e termina con il tag <HTML></HTML>.
Nel corso degli anni si è assistito a diverse versioni di questo linguaggio di markup, l'ultima, utilizzata anche per creare la struttura della *web application*, è la versione 5.

4.2 CSS

CSS (*Cascading Style Sheets*), chiamati anche semplicemente fogli di stile, è un linguaggio utilizzato per descrivere la presentazione visuale di un documento HTML.
Esso è stato introdotto nel 1996 dal World Wide Web Consortium[3], anche conosciuto come W3C, un'organizzazione che ha come scopo quello di sviluppare tutte le potenzialità del World Wide Web.
Mediante l'uso dei fogli di stile CSS è possibile definire, attraverso delle semplici regole, il modo in cui debbano essere presentati alcuni elementi di un documento HTML come il colore, la dimensione e lo stile del testo, ma anche elementi più complessi come il layout di pagina, le sfumature, etc. Una regola CSS segue la seguente sintassi:

```
selettore {proprietà: valore};
```

Esempio:

```
p {color: red};
```

Dove per ***selettore*** si intente un'espressione secondo il formato previsto da CSS, il risultato di tale espressione è una serie di elementi (in genere HTML ma si può estendere anche a qualsiasi tag) a cui dover modificare la (o le) ***proprietà*** aggiornandola con il ***valore*** fornito.

I fogli di stile possono essere applicati ad un documento HTML in tre modi diversi:

- **Stili inline:** lo stile viene applicato direttamente all'elemento HTML;

[3] https://www.w3.org/Consortium

- **Fogli di stili incorporati:** il foglio di stile viene implementato a inizio documento HTML mediante il tag <style> che deve comparire all'interno dell'intestazione;
- **Fogli di stile esterni:** gli stili vengono specificati su un file distinto. La separazione dello stile dalla struttura del testo è comunemente ritenuta una buona pratica in quanto è possibile applicare lo stesso stile a tutte le pagine che fanno riferimento a esso con una sola operazione.

Per la realizzazione dello stile delle pagine HTML si è utilizzato CSS3 (l'ultima versione) su un file separato, collegando quest'ultimo a tutte le pagine del progetto.

4.3 Bootstrap

Bootstrap è un ambiente di lavoro o *framework* per lo sviluppo web, fu inizialmente creato nel 2010 da due sviluppatori che lavoravano per Twitter. Nel giro di pochi mesi fu adottato da Twitter per la creazione di strumenti utilizzati internamente all'azienda, circa un anno dopo fu rilasciato al pubblico[4]. Attualmente l'ultima versione è Bootstrap 3, la stessa versione utilizzata nel progetto.
La missione del suddetto strumento è quella di semplificare lo sviluppo web per quanto riguarda la creazione di elementi comuni nelle interfacce grafiche. Elementi quali bottoni, menu a tendina, form, tabelle o messaggi di avviso trovano già uno schema di base con Bootstrap; sarà poi il programmatore a compiere tutte le modifiche del caso affinché rispecchino la sua volontà.
Esistono altri due importanti motivi per utilizzare Bootstrap:
Il primo riguarda l'accessibilità (vedi Sezione 5.2*)*, questo *framework* o ambiente di sviluppo facilità la creazione di siti web *responsive.* In un mondo sempre più dominato dall' IoT (*Internet of things*) risulta cruciale riuscire a creare un sito che possa adattarsi bene a qualsiasi piattaforma utilizzata dall'utente per accedere al sito. Il suddetto ambiente di sviluppo viene incontro alle necessita dei *front-end developers* (vengono designati in questo modo gli sviluppatori web che si occupano di interfacce grafiche direttamente esposte all'utente) andando a offrire componenti grafici pensati per poter scalare al meglio in relazione allo spazio disponibile, spazio che sarà dettato dallo schermo che l'utente utilizza per accedere al sito, che sia un computer, un tablet o un telefono Bootstrap permette ottimi compromessi tra spazio disponibile e accessibilità del sito.

[4]http://getbootstrap.com/about/

Anche il secondo motivo è sempre inerente accessibilità e usabilità ma con una accezione leggermente differente, il focus è sempre sull'esperienza grafica indirizzata all'utente ma invece che avere sotto esame la parte di design del sito bisogna tenere in considerazione il codice prodotto pe arrivare all'obiettivo finale. Infatti non tutti i browser supportano le stesse istruzioni CSS o codice HTML o almeno non è detto che tutte le versioni di un determinato browser supportino una data istruzione. Ad esempio, durante la creazione di una transizione CSS che cambia gradualmente la larghezza di un oggetto bisogna tenere in conto che Safari, il browser di Apple segue una sintassi diversa:

```
-webkit-transition: width 2s linear 1s; /* For Safari 3.1 to 6.0 */
 transition: width 2s linear 1s;
```

Nel caso si ometta l'istruzione CSS per Safari, gli utenti che accedono il sito da Mac non saranno in grado di visualizzare tale animazione. Il codice riportato sopra è solo un esempio da manuale ma possono presentarsi tanti piccoli artefatti dovuti a differenti implementazioni di comandi CSS o dovuti alla mancanza di supporto di determinati tag HTML. Bootstrap è molto utile per questo, esso riesce a appianare al meglio le divergenze che sorgono dall'uso di un browser a un altro.
Il risultato sarà quindi un sito che riesce a scalare bene su tutti gli schermi e che avrà una visualizzazione consistente indipendentemente dal browser utilizzato.

4.4 Il protocollo HTTP

Il sito poggia sul protocollo HTTP, per lo scambio di dati tra client e server, il client quindi si occupa di gestire l'interazione tra il sito e l'utente andando quindi a gestire l'inserimento dei dati da parte dell'utente, guidandolo verso la fase di creazione di *dataset* oppure gestendo il caricamento/scaricamento di *dataset*. Mentre il server è delegato a ricevere dal client i dati che l'utente ha inserito e di processarli secondo la volontà dell'utente finale, al termine della computazione il server fornirà una risposta al client sotto forma di testo o immagini, per lo più grafici. Come ultima fase il client dovrà mostrare a video i risultati della computazione.

4.4.1 Breve introduzione

Sono le fondamenta su cui si regge l'intera comunicazione e trasmissione di dati all'interno del World Wide Web. I protocolli HTTP e HTTPS, creati a cavallo tra il 1989 e il 1994 da, rispettivamente, Tim Berners-Lee e Netscape Communications, svolgono il ruolo di mediatore all'interno del modello client-server e permettono lo scambio di informazioni tra due nodi della Rete gestendo sessioni di comunicazione, richieste e quant'altro connesso a questo processo.
Nato in concomitanza con la nascita del web moderno, il protocollo HTTP (Hyper-Text Transfer Protocol, "protocollo di trasmissione di documenti ipertestuali") funziona come un protocollo di richiesta-risposta all'interno del modello client-server. Un esempio classico di ciò è quello di un web browser, esso svolge la funzione di *client* ovvero di cliente, lo scopo del browser è quello di collegarsi a un'altra macchina connessa in rete mediante il meccanismo delle URL; la macchina a cui il browser si collega è detta *server*, il suo scopo è quella di accogliere richieste provenienti da altri *client* e di fornire loro i dati richiesti.

Le risorse HTTP sono localizzate e identificate nel web grazie al sistema di *Uniform Resource Locator* (URL), ovvero il comune "indirizzo web".
Una URL è principalmente composta da 5 parti:

- **Nome del protocollo internet usato:** Specifica se usare HTTP o la sua versione sicura HTTPS
- **Nome del dominio a cui collegarsi:** È un identificatore univoco volto a mappare un sito presente sul Web
- **Path:** Percorso che identifica una risorsa all'interno della macchina server
- **Query:** È possibile che una risorsa venga generata dinamicamente in base a una certa domanda. Frequente è il caso in cui ci si colleghi a siti che fanno uso di database. Questa parte della URL è opzionale.
- **Parametri:** Essi servono a specificare le parti della query interessate e seguono la sintassi `<nome_parametro> = <valore_parametro>`.
  Questa parte della URL è opzionale.

Un esempio di URL completa delle parti sopra citate. La prima parte riguarda il collegamento ai server di YouTube, poi viene richiesta la pagina dei risultati di ricerca e come parametro di ricerca viene specificato che il nome dei video da ricercare deve contenere la stringa "Vasco".

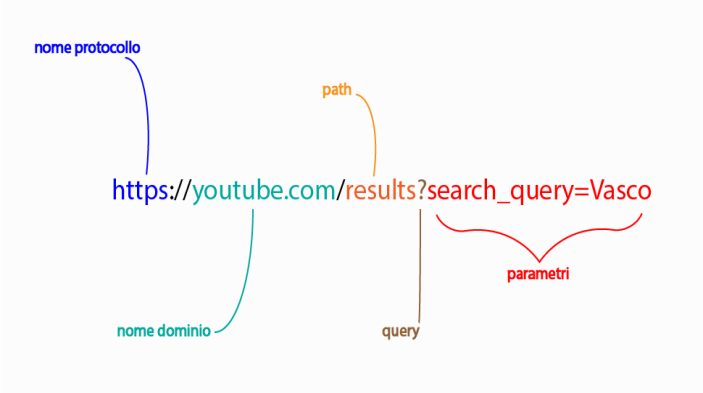


Figura 13. Esempio URL[5]

4.4.2 Funzionamento di una sessione http

Quando *client* e *server* si mettono in contatto l'uno con l'altro si stabilisce una sessione HTTP. Si tratta di una sequenza di richieste di rete tra i due nodi, grazie alla quale l'uno (il *client*) chiede e ottiene informazioni dall'altro (il server).
Il server dispone di più canali di comunicazioni attivi simultaneamente per poter riuscire a offrire più servizi contemporaneamente a più client attivi, questi canali sono detti *porte*. La porta usata per il protocollo HTTP è la numero 80, mentre ad esempio esiste la porta 110 volta alla scaricamento di e-mail; è quindi possibile creare un server che mentre fornisce a un client una pagina web permette a un altro utente di scaricare i messaggi di posta elettronica.
Una sessione ha inizio quando il *client* stabilisce una connessione con una particolare *porta* del server (numero 80) inviando una richiesta di informazioni o risorse. Il server risponde poi con una linea di stato (del tipo: “HTTP1.1 200 OK”) e un messaggio contenente le informazioni richieste o, eventualmente, un messaggio di errore.

[5] La seguente immagine è stata creata usando la URL restituita a seguito di una ricerca su YouTube

4.4.3 Metodi di richiesta del protocollo HTTP

Affinché lo scambio di informazioni vada a buon fine, il *client* deve inviare le proprie richieste seguendo una sintassi specifica. Per ogni azione base che una macchina può compiere è stato definito un verbo.

Tra i verbi più comuni:

- ***GET:*** rappresenta la richiesta di invio di una specifica risorsa ospitata sul server. Con questo comando è possibile recuperare il dato ma non compiere altre operazioni (come ad esempio cancellare o modificare il dato sul server)
- ***HEAD:*** stessa funzionalità di GET ma senza ottenere l'intero corpo della risposta limitandosi alle sole informazioni riassuntive presenti nell'intestazione. Utile nel caso si debbano recuperare delle meta-informazioni scritte nello *header* ("intestazione"), quali tipo di risorsa (immagine/documento/ecc.), data creazione, data ultima modifica di una risorsa web senza essere costretti a scaricare l'intero documento.
- ***POST:*** richiede al server di accettare l'entità allegata al messaggio in modo tale da poter creare una risorsa sul server stesso a partire da ciò che il client ha spedito al server. Un esempio comune può essere la creazione di un commento su un forum; il testo del commento sarà la risorsa che il client chiederà al server di salvare in modo tale che altri utente possano vederla.
- ***DELETE:*** richiesta di cancellazione o eliminazione di una specifica risorsa dal server

4.5 jQuery

Per quanto riguarda la parte la to client è stata usata la libreria Javascript jQuery[6][7].
Il suddetto software nasce con lo scopo di semplificare la manipolazione degli elementi che compongono la pagina, facilitare l'inserzione di animazioni, gestione degli eventi e delle richieste da effettuare al server. Tutte le caratteristiche sopra citate vengono garantite compatibili con i principali browser in modo tale da garantire lo stesso comportamento indipendentemente dal browser che l'utente usa.
Sempre inerente a jQuery c'è stata l'adozione di un plugin[8] volto a creare una piccola schermata di attesa classica per accompagnare i caricamenti nel caso richiedano

[6]https://jquery.com/
[7]https://it.wikipedia.org/wiki/JQuery

qualche secondo, tipicamente è il caso in cui vengono eseguiti gli algoritmi che necessitano di un po' di tempo per l'elaborazione dei risultati.
Mentre per generare alcuni messaggi di avviso o di errore in maniera dinamica la scelta è ricaduta sulla libreria Bootbox[9], la quale poggia sia su jQuery, sia su Bootstrap. Tale libreria permette di creare finestre da mostrare all'utente in maniera molto veloce e con buone possibilità di personalizzazione. Nella sua forma base l'istruzione per creare un avviso per l'utente è la seguente:

```
bootbox.alert("This is the default alert!");
```

Il risultato prodotto è il seguente:

Figura 14. Esempio utilizzo Bootbox

4.5.1 Applicazioni pratiche

jQuery risulta molto utile per quanto riguarda la manipolazione degli elementi presenti in una pagina HTML. Ogni comando che fa uso della suddetta libreria deve iniziare con il simbolo del dollaro o con 'jQuery', la sintassi è la seguente:

```
($ | jQuery).(selettore).azione()
```

Dove ***selettore*** rappresenta l'espressione con cui trovare gli elementi, ***comando*** è una qualsiasi funzione della libreria

Si consideri la seguente istruzione jQuery:

```
$("p.test").hide();
```

Il selettore "p.test" segue la stessa sintassi dei selettori CSS, seleziona quindi all'interno della pagina tutti i paragrafi di testo (p) di classe test (.test). A questi paragrafi trovati applica la funzione hide che serve a nascondere i paragrafi dalla pagina. In Javascript avremmo dovuto scrivere queste istruzioni per eseguire questo semplice compito:

[8] https://gasparesganga.com/labs/jquery-loading-overlay/
[9] http://bootboxjs.com/

```
var paragrafiNellaPagina = document.getElementsByTagName("p");
for(var c=0; c< paragrafiNellaPagina.length; c++){
    if(paragrafiNellaPagina[c].getAttribute("class") === "test"){
        paragrafiNellaPagina[c].style.display = "none"
    }
}
```

Andiamo a recuperare l'elenco dei paragrafi, controlliamo ogni paragrafo per vedere se possiede la classe 'test', se si lo nascondiamo dalla pagina in maniera dinamica. Come si può vedere il risparmio di codice è notevole, 1 riga contro 6 anche nei compiti più semplici e comuni. Ciò permette al programmatore di risparmiare tempo e codice nell'esecuzione di molte operazioni; scrivere meno codice significa anche diminuire il rischio di introdurre bug nel proprio codice, i quali che potrebbero portare a comportamenti inaspettati e difficilmente replicabili.
jQuery non si limita solo a manipolare gli elementi di una pagina web ma offre anche gli strumenti per effettuare richieste asincrone a un server arbitrario (vedi Sezione 4.4.2). Quindi nel progetto le richieste effettuate al server sono state eseguite per mezzo di jQuery.
La sintassi di base per effettuare tali richieste è la seguente[10]:

```
$.ajax({
        url: <URL>,
        {
            data: <JSON_Array>,
            success: <Result_Callback>
      }
});
```

Dove <URL> è una generica URL riferita a un server, <JSON_Array> è un oggetto JSON in cui i dati sono salvati come coppia id-valore. Questo oggetto rappresenta l'insieme dei parametri che il server potrà usare per elaborare la richiesta. Infine, <Result_Callback> è una funzione avente come parametro i dati ricevuti dal server, è all'interno di questa funzione che si decide cosa fare con i dati ricevuti dal server.

[10]http://api.jquery.com/jQuery.ajax/

Da notare che nell'esempio **data** e **success** sono solo due delle impostazioni che si possono specificare, ne esistono molti altri che permettono di ottenere un maggior controllo sulle modalità delle richieste da inviare e sul tipo di dati attesi.
Nel progetto ad esempio come URL è stata usata la seguente URL per scegliere di disegnare un *dataset* usando l'algoritmo Kmeans:

/KMeans/draw_dataset

I possibili parametri passati sono l'insieme di punti su cui l'algoritmo dovrà operare e le opzioni dell'algoritmo corrente. Il server usa questi dati appena citati per eseguire l'algoritmo, i dati restituiti sono una pagina web contenente dei grafici e del testo a corredare i grafici.

All'interno del progetto esistono anche altre URL specializzate in altri compiti ad esempio:

- ***/Kmeans/generate_numeric_dataset:*** si occupa di generare un *dataset* valido per l'algoritmo Kmeans. Restituisci quindi un insieme di punti ciascuno composto da coppie di valori X-Y.
- ***/Dbscan/generate_numeric_dataset:*** creare un *dataset* per l'algoritmo Dbscan. Questa route condivide lo stesso codice della URL sopra citata, si è scelto lo stesso di creare una URL apposita nel caso di eventuali espansioni future.
- ***/DecisionTree/options:*** restituisce una pagina HTML inerente le impostazioni dell'algoritmo Decision Tree.

4.6 Web Framework

Per la parte di programmazione lato server, la scelta è ricaduta sui *Web framework*; conosciuti anche solo come '*web application framework*', sono ambienti di lavoro volti a rendere semplice la scrittura e il mantenimento del software man mano che esso cresce con il passare del tempo. I *Web framework* includono strumenti che semplificano le comuni operazioni di sviluppo web quali, gestione degli URL e dei database, gestiscono l'output sotto forma di pagina web e includono anche miglioramenti per quanto riguarda la protezione da attacchi online.
Nel progetto per la parte relativa al server è stato usato il *web framework* Flask, il quale è basato sul linguaggio di programmazione Python, risulta essere quindi com-

patibile con il risolutore di esercizi fornitomi. Il suo compito è quello di eseguire la libreria software che mi è stata fornita dai docenti, libreria che è in grado di risolvere tipi di esercizi di Data Mining che gli studenti saranno poi chiamati a risolvere loro stessi durante l'esame.

4.6.1 Il paradigma MVC

Uno dei vantaggi principali nell'utilizzo di un *web framework* rispetto a un server tradizionale è il supporto al pattern architetturale MVC. Nel paradigma *MVC* (*Model View Controller*), si fa un'attenta distinzione tra i dati sottostanti l'applicazione (*model*) e le modalità con le quali essi vengono presentati all'utente (*view*), l'interazione tra le due parti è gestita da un componente separato (*controller*).
Tra i principali vantaggi della separazione della logica applicativa dalla presentazione troviamo una maggiore indipendenza tra i componenti coinvolti e una più semplice manutenibilità dell'applicazione: per esempio, diventa possibile apportare modifiche alla vista senza intervenire sul modello, ma anche avere viste differenti di uno stesso modello di dati senza necessariamente intervenire sulla vista.
Avremo quindi un utente che fa una richiesta al *controller* del server, il *controller* va a recuperare un modello o template preconfezionato da riempiere con i dati ricevuti dal *controller*. Dall'unione di dati e template deriva la costruzione della *View*, ovvero la vista finale del risultato da spedire indietro e che poi verrà presentato all'utente.

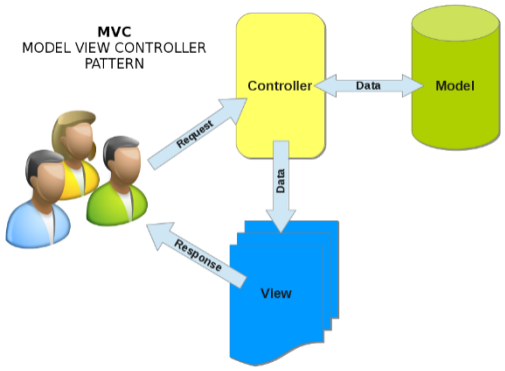


Figura 15. Modello MVC

4.6.2 Altri vantaggi di un web framework rispetto a un server classico

I *web server*s e i browser comunicano direttamente usando il protocollo HTTP: il server rimane in attesa di richieste HTTP dal browser, successivamente elabora le richieste e alla fine restituisce al browser una risposta HTTP. I *Web framework* facilitano il compito sollevando l'utente dal dover scrivere codice che si occupa di interagire con le richieste HTTP a basso livello, in quanto è il *web framework* stesso a prendersi questa responsabilità.
Il *web framework* usato nel progetto è Flask, esso è basato sul linguaggio di programmazione Python. Il motivo per cui è stato scelto è perché la libreria software in grado di risolvere gli esercizi è anch'essa scritta in Python, inoltre Flask è molto famoso per la sua leggerezza, aspetto che consente di installare il server anche su macchine non particolarmente prestanti.
La maggior parte dei siti offrono un gran numero di risorse, il cui numero è destinato a crescere nel tempo, queste risorse sono accessibili tramiti delle URL. I *web framework* integrano quindi in maniera nativa un meccanismo per associare a una determinata URL a una procedura, la quale sarà delegata a offrire una certa risorsa. Questo livello di astrazione appena introdotto (URL-procedura) è molto utile in termini di manutenibilità del codice in quanto un cambiamento nel nome della URL non si traduce in un bisogno di cambiare anche il codice relativo a quella data URL.
Alle URL definite in un *web application framework* possono essere associati anche degli argomenti/parametri per differenziare la logica di generazione dei contenuti andando quindi incontro a una maggiore dinamicità dei contenuti generati.

Il seguente frammento definisce la URL principale del sito (per convenzione '/'), alla URL è associata una funzione chiamata 'hello' la quale prende come argomento una stringa di testo che identifica un nome e restituisce come output una pagina HTML con un paragrafo dal contenuto variabile in base al nome utente.

```
@app.route("/")
def hello():
    nomeUtente = request.args.get('nome_utente')
return '<p> Ciao ' + nomeUtente + '</p>'
```

Si noti che quello che il precedente frammento di codice restituisce effettivamente è solo un paragrafo (tag <p> in HTML), è il framework stesso a prendersi cura di crea-

re una pagina web completa di tutte le sue parti e di inserire al suo interno il paragrafo, ciò permette allo sviluppatore di non doversi curare di impiegare tempo per scrivere sempre una piccola pagina HTML, un altro vantaggio oltre al risparmio di tempo è anche relativo alla pulizia del codice scritto che risulta essere molto più chiaro e coinciso.
C'è un ultimo grande punto di forza nell'uso di un web framework, abbiamo detto in precedenza che i server devono fornire risorse sotto forma di risposte HTTP, queste risorse possono essere qualunque cosa, testo, immagini, audio.
Con i web framework vengono anche distribuiti dei template engine, ovvero degli pseudo-parser HTML, essi hanno il compito creare una normale pagina HTML facendo uso di costrutti tipici di linguaggi di programmazione quali IF-ELSE/FOR-LOOP e funzioni ausiliarie varie come manipolazione di stringhe, numeri o creazione di variabili.
Ciò che prima era possibile fare con macchinose istruzioni rendendo il codice più difficile da leggere e/o scrivere e da modificare nel caso ci fossero anche dei minimi cambiamenti alla struttura della risorsa che si voleva restituire al client, adesso è possibile farlo in tutta semplicità.

Si riporta un esempio di *template* HTML "aumentato" con l'aggiunta di comuni costrutti dei linguaggi di programmazione. Il codice riportato è compatibile con il *template engine* fornito con Flask, ovvero Jinja2

```
{% if fruit_list %}
    <ul>
      {% for fruit in fruit_list %}
            <li>{{ fruit }}</li>
      {% endfor %}
    </ul>
{% else %}
      <p>No fruits are available.</p>
{% endif %}
```

Ciò che viene generato in output è una lista di nomi di frutta, se invece non è disponibile nessuno frutto viene restituita una pagina con un paragrafo che avvisa l'utente.

5 Progettazione e grafica

Negli ultimi anni, i software disponibili sono cresciuti in complessità e raffinatezza e hanno raggiunto un pubblico sempre meno tecnico. Ciò ha inevitabilmente spostato la progettazione e lo studio dell'interfaccia utente (UI) in una posizione di rilievo: infatti, per essere realmente utili, i nuovi strumenti digitali devono essere intuitivi, facili da usare e accessibili, così da poter essere adoperati dal più alto numero di persone. Risulta quindi fondamentale iniziare a curare maggiormente la progettazione dell'interfaccia grafica; deve essere semplice, chiara e intuitiva; deve mettere l'utente in condizione tale da sapere subito cosa fare, cosa cliccare, cosa aspettarsi; deve infine essere piacevole e ispirare affidabilità.

5.1 Struttura del sito

Il sito viene presentato all'utente come se fosse una singola pagina, esso in realtà viene costruito man mano che l'utente effettua richieste.
Si parte da uno scheletro generale contenuto nella pagina index.html, qua viene definita, l'intestazione, il menu e un possibile footer. In base a quale voce del menu l'utente sceglie viene caricata un'altra pagina; essa viene inserita nel template in un apposito contenitore.
In base a quale algoritmo l'utente sceglie, possono essere scaricati due template:

1. Template per algoritmi che operano con dati numerici per Kmeans, Dbscan, Hierarchical
2. Template per algoritmi che operano con dati testuali per Decision Tree e Apriori

Per gli utenti più esperti o che hanno delle difficoltà su un certo tipo di algoritmo è possibile accedere alla relativa pagina direttamente tramite URL da specifica, ad esempio:

```
<Dominio del sito>:<numero di porta>/K-Means
```

Questo farà sì che l'utente non vedrà la pagina di aiuto iniziale, verrà invece portato subito alla pagina specifica di K-Means, velocizzando cosi la navigazione.
Inoltre, quando l'utente va a configurare le impostazioni di un certo algoritmo il client effettua una richiesta di un altro template al server. Una volta ricevuta la pagi-

na la riempie con i valori delle impostazioni correnti e la presenta all'utente nel caso voglia apportare qualche modifica.

Per esempio, se l'utente vuole accedere alla pagina delle impostazioni dell'algoritmo Decision Tree, il client, come visto nel capitolo inerente jQuery (vedi Sezione 4.5) effettua una richiesta al server sulla seguente URL:

/DecisionTree/options

La quale restituisce una pagina HTML che una volta inviata al client viene inserita in un contenitore che appare "sovrapposto al sito" (vedi Figura 16. Finestra per la modifica delle opzioni per Decision Tree). Si è optata questa scelta per riuscire a creare un sito molto intuitivo e grazioso, senza i fastidiosi ricaricamenti della pagina dovuti al continuo cambio di pagina. Inoltre, anche se le pagine richieste sono al massimo pochi KB, le suddette pagine, una volta richieste vengono mantenute in memoria fino alla chiusura del sito per limitare le richieste al server.

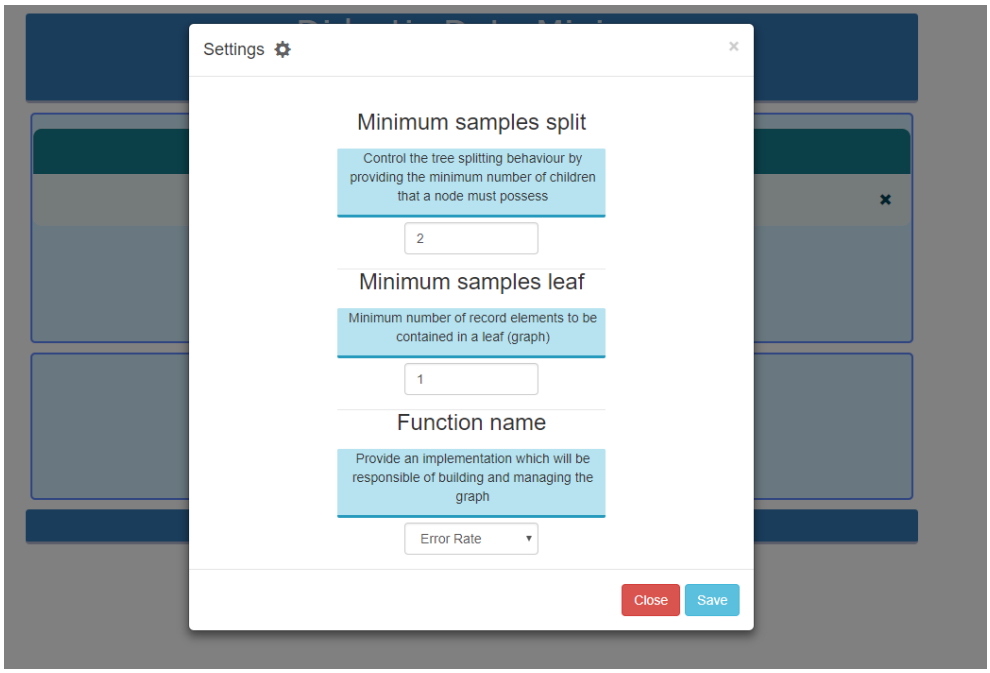


Figura 16. Finestra per la modifica delle opzioni per Decision Tree

È garantita inoltre un alto grado di manutenzione, in quanto si è scelto di "spezzare" il sito su varie pagine, facilitando così la modifica dei singoli componenti del sito.

Se ad esempio ci fosse necessità di aggiungere una nuova opzione basterebbe modificare il file delle opzioni per un determinato algoritmo e aggiungere il codice Javascript per gestire anche la nuova opzione selezionata. Non si hanno più cosi file enormi, pieni di contenuti da dovere gestire, ma piccoli file specializzati.

5.2 Accessibilità e usabilità

Quando si progettano e sviluppano applicazioni interattive, uno degli obiettivi principali è quello di renderle usabili per gli utenti finali. Lo standard W3C ISO 9241 definisce l'usabilità come «la misura in cui un prodotto può essere usato da specifici utenti per raggiungere determinati obiettivi con efficacia, efficienza e soddisfazione, in uno specifico contesto d'uso»[11]. Un sistema si dice, dunque, usabile quando rende efficiente e soddisfacente l'esperienza dell'utente che lo sta visitando e risponde ai suoi bisogni informativi, fornendogli facilità di accesso e navigabilità, e consentendogli un adeguato livello di comprensione dei contenuti. L'usabilità risulta molto importante in quanto aumenta l'efficienza, limita gli errori e riduce il bisogno di addestramento degli utenti, che quindi accettano più volentieri l'uso di applicazioni informatiche.

Un altro tema da tener in considerazione durante la progettazione di siti web è quello dell'accessibilità; un sistema viene considerato accessibile quando può essere visitato da qualsiasi utente indipendentemente dal dispositivo usato, dalla velocità del collegamento internet, dal browser, dalla lingua o cultura dell'utente, dalla sua posizione e dalle sue capacità fisiche o mentali.

Nei sistemi accessibili, il contenuto dell'informazione e la sua presentazione sono solitamente indipendenti l'uno dall'altra, in modo da permettere una riorganizzazione dei contenuti sulla base delle esigenze reali dell'utente.

Usabilità e accessibilità sono strettamente correlate, ma non sono la stessa cosa: l'accessibilità è volta ad allargare il numero degli utenti, mentre l'usabilità è volta a rendere gli utenti più efficienti e soddisfatti.

Può esserci, infatti:

- **usabilità senza accessibilità:** il sistema è perfettamente usabile, ma vi sono fasce di utenti che non riescono ad accedervi;

[11] Cfr. «**Usability:** extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.» da *ISO 9241- Guidance on usability*: https://www.iso.org/obp/ui/#iso:std:16883:en.

- **accessibilità senza usabilità:** tutti gli utenti riescono ad accedervi ma alcuni di essi incontrano difficoltà perché il sistema non è usabile.

5.3 Principi generali di sviluppo

Una buona interfaccia utente (abbreviato GUI) è quella invisibile che non crea ostacoli al reperimento dei contenuti; deve essere progettata in modo da garantire la massima efficienza e semplicità d'uso all'utente.
Un'interfaccia grafica deve innanzitutto permettere all'utente di essere consapevole della situazione, ovvero capire cosa sta succedendo per mezzo di opportuni feedback e percepire correttamente gli elementi e il loro significato.
In ogni caso, è necessario fare in modo che gli utenti riconoscano agevolmente gli elementi e capiscano subito come questi possono essere utilizzati.
L'interfaccia deve inoltre evitare di presentare informazioni, gli elementi devono essere selezionati attentamente e le modalità di presentazione devono essere semplici e chiare.
Infine, una buona GUI deve essere in grado di adattarsi alle risorse disponibili nell'ambiente in cui viene utilizzata e funzionare correttamente in ogni situazione.
La diversità sempre maggiore dei dispositivi e delle piattaforme, attraverso le quali gli utenti possono accedere ad un sistema, rende necessaria una cura più attenta dell'interfaccia anche in termini di fluidità e *responsiveness*, ovvero di adattamento ed eventuale riorganizzazione degli elementi.

5.4 Risultato

Nel progetto si è scelto quindi di usare uno stile minimale volto a facilitare e rendere più veloce possibile il lavoro degli studenti che andranno a usare la suddetta *web application.* Anche la scelta del font Helvetica[12] contribuisce a tale scopo, questo carattere fa parte della categoria dei font senza grazie, fu introdotto per la prima volta nel 1957. Il suo successo crebbe a tal punto che fu adottato per la segnaletica della città di New York, i motivi principali di tale successo riguardano l'eleganza dei caratteri, che con la loro essenzialità unita a un alto grado di leggibilità e al loro grado di neutralità rendono questi caratteri un'ottima scelta in generale. Si è poi scelto di accompagnare il caricamento dei contenuti con delle leggere e brevi animazioni a compar-

[12]https://it.wikipedia.org/wiki/Helvetica

sa. L'obiettivo di queste animazioni è quello di rendere più piacevole la navigazione del sito senza però andare a appesantire o rallentare il sito per la continua attesa della fine di lunghe e invadenti animazioni. Per quanto riguarda le *palette* di colori usata, la scelta è stata quella di utilizzare principalmente il blu e le sue tonalità in quanto secondo studi sembra che il blu sia il colore più rilassante per la vista.

Mentre per garantire il massimo grado di accessibilità del sito è stato usato il framework Bootstrap (vedi Sezione 4.3) la cui missione è quella di aiutare l'utente nel creare siti web ottimali per ogni piattaforma (sia essa un computer, telefono, televisore) garantendo così il massimo grado di efficienza nella riorganizzazione del layout per poter usare al meglio lo spazio disponibile. Il sito risulta essere cosi perfettamente usabile sia da computer che da *mobile* (*mobile friendly*).

La schermata iniziale del sito (vedi Figura 17. Schermata del sito iniziale) mostra una breve guida, un sunto delle operazioni salienti che è possibile svolgere nel sito, poi il menu del sito guida l'utente attraverso la navigazione permettendo la scelta dell'algoritmo da usare per risolvere l'esercizio.

Ogni algoritmo opera su un certo insieme di valori/dati detti *dataset*, i *dataset* disponibili per l'algoritmo scelto sono elencati sotto forma di tabella; cliccando su ciascuna voce della tabella verranno mostrati i valori contenenti; è possibile modificare o rimuovere i punti già esistenti oppure se ne possono aggiungere di nuovi. Viene poi inclusa la possibilità di caricare *dataset* direttamente da file, utile per importarne svariati tutti insieme ed è disponibile anche il download su file dei *dataset* correnti.

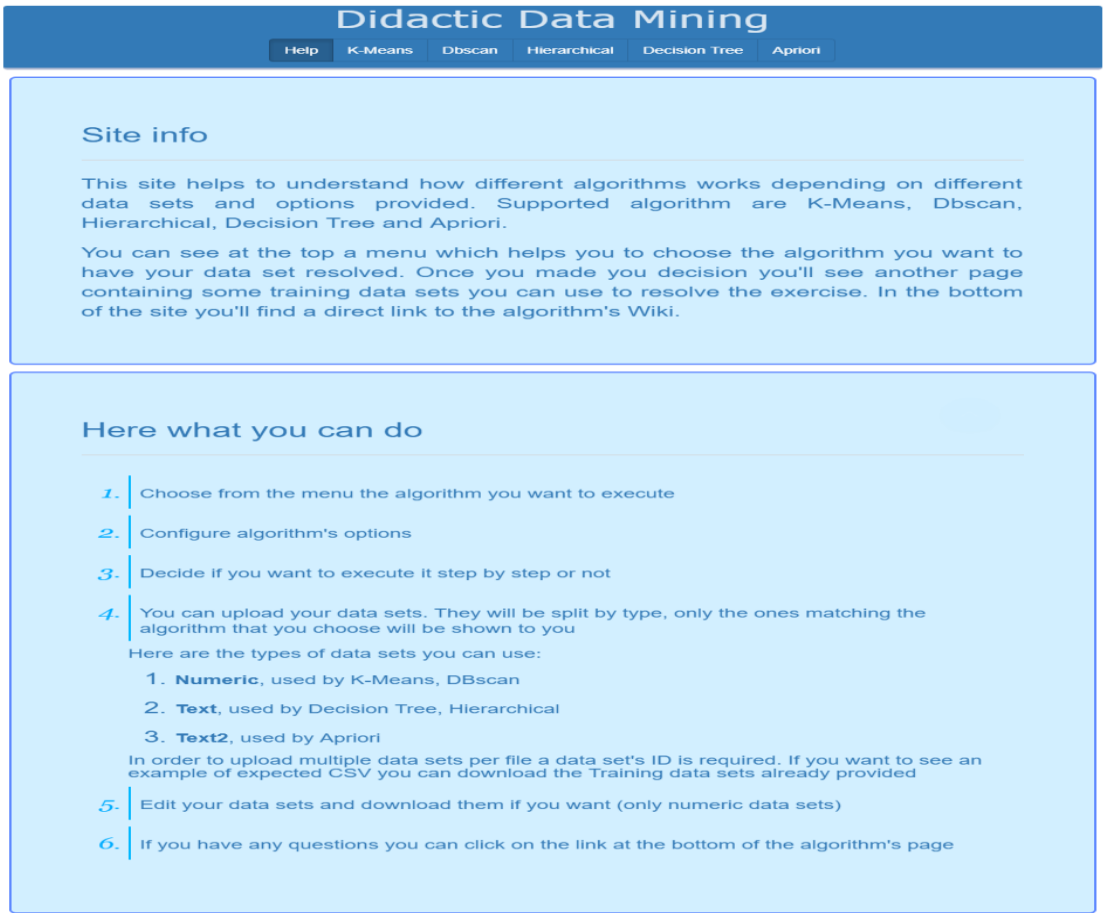


Figura 17. Schermata del sito iniziale

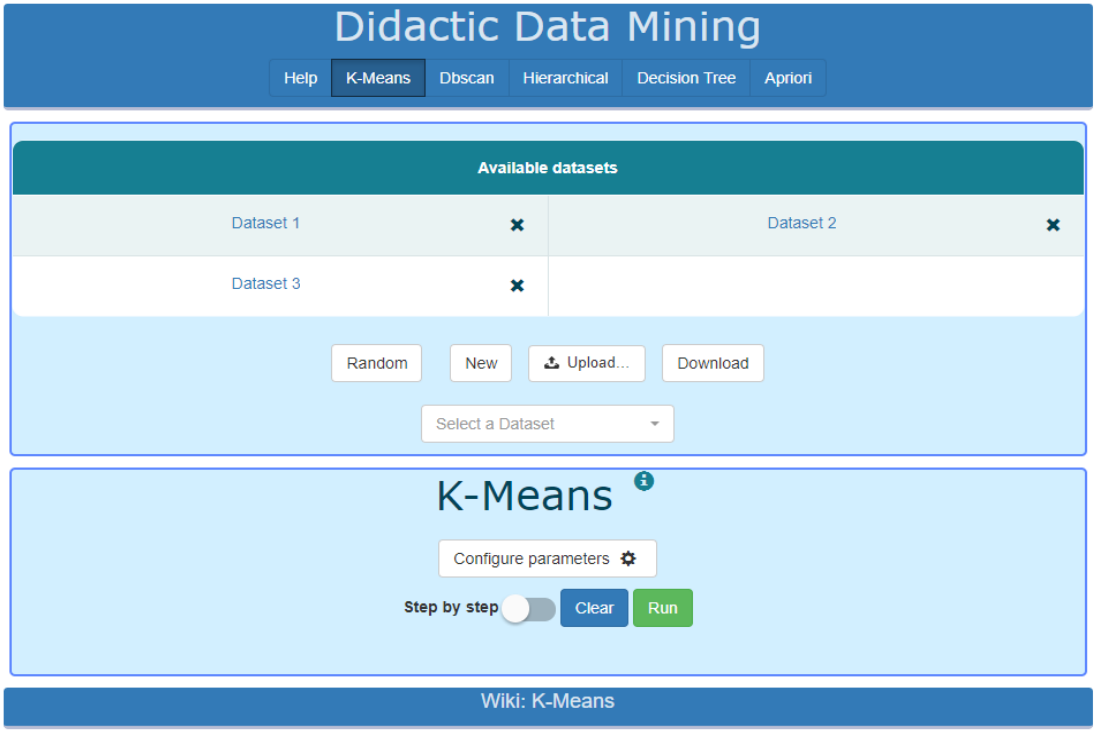


Figura 18. Schermata del sito di K-Means

L’utente dopo aver selezionato una delle voci di menu (vedi immagine sopra) si trova davanti due sezioni logicamente distinte:

- Sezione inerente i dati:

  Viene presentata all’utente una tabella con i dati disponibili (*datasets*) dove può scegliere di eliminarli o di modificarli con un click sul *dataset* desiderato. Per la modifica dei dati compare una finestra in sovraimpressione (vedi Figura 19. Finestra modifica dataset) in cui l’utente può cambiare il valore dei punti del *dataset*, può anche rimuovere le singole coordinate; nel caso di input non corretto viene evidenziato il punto in cui l’utente ha commesso un errore con relativo messaggio.

  È possibile poi creare nuove collezioni di dati automaticamente premendo sul pulsante “*random”* oppure è possibile farlo manualmente tramite il bottone “*new”*. I bottoni “*Upload*” e “*Download*” servono per importare o esportare i dati.

- Sezione relativa all’algoritmo scelto:

  Sono presenti i controlli per la risoluzione degli esercizi quali attivare o disattivare la modalità passo-passo, cancellare la soluzione ottenuta oppure iniziare a produrre una soluzione (In ordine *Step by step*, *Clear, Run*).

  Inoltre è presente anche un piccolo pulsante informazioni, che se premuto fornisce all’utente una piccola spiegazione sul funzionamento del suddetto algoritmo; nell’immagine soprastante (Figura 18. Schermata del sito di K-Means) accanto alla scritta K-Means, è possibile osservare tale bottone.

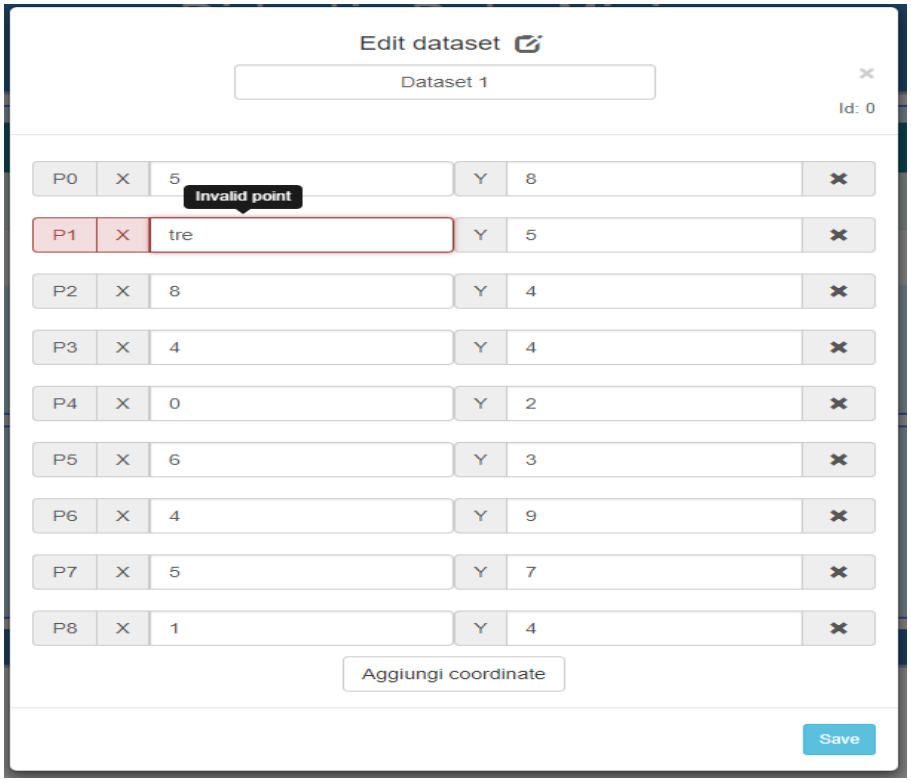


Figura 19. Finestra modifica dataset

L'utente può configurare le opzioni dei singoli algoritmi in modo da studiarne il comportamento al variare di ogni parametro. Ogni opzione disponibile è accompagnata da una breve spiegazione per facilitare l'utente.

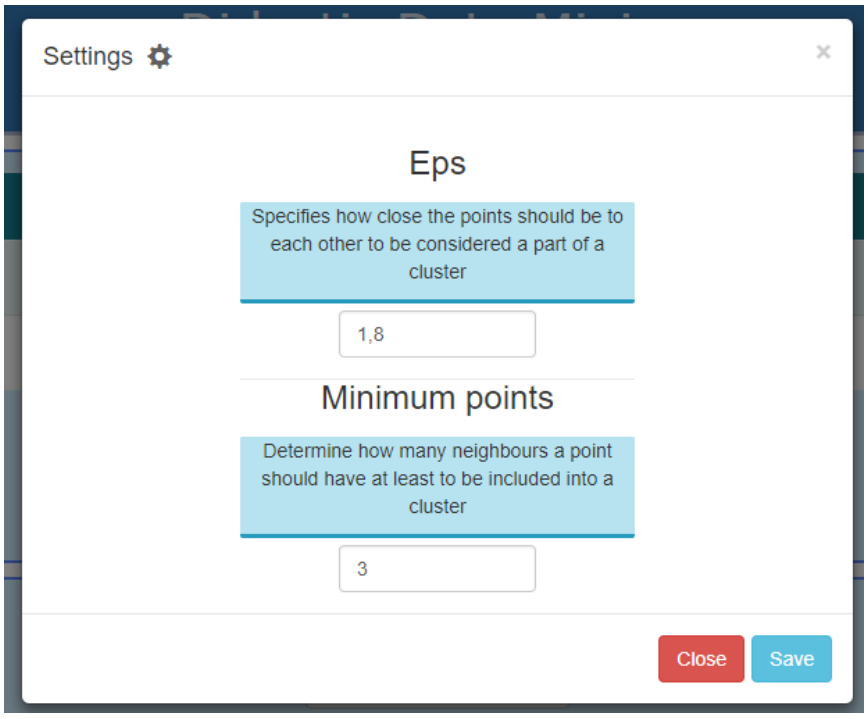


Figura 20. Finestra per la modifica delle opzioni per Dbscan

Al momento dell'esecuzione, è possibile specificare se la soluzione deve essere passo-passo o se deve essere presentata nella sua completezza. L'immagine sotto mostra K-Means in esecuzione (vedi Figura 21. Schermata del sito con K-Means in esecuzione). Si può notare che anche scorrendo la pagina verso il basso rimane comunque visibile la barra relativa alle impostazioni dell'algoritmo che comprende il bottone che dà informazioni generali sull'algoritmo e i comandi per chiedere una nuova computazione, utile per chiedere velocemente una nuova soluzione dopo aver cambiato qualche parametro. Sempre nella foto è stata chiesta una soluzione intera e ogni immagine rappresenta una iterazione, un passaggio che K-Means compie. Sempre in basso un'immagine di K-Means in esecuzione passo-passo (vedi Figura 22. Schermata del sito con K-Means in esecuzione passo-passo).

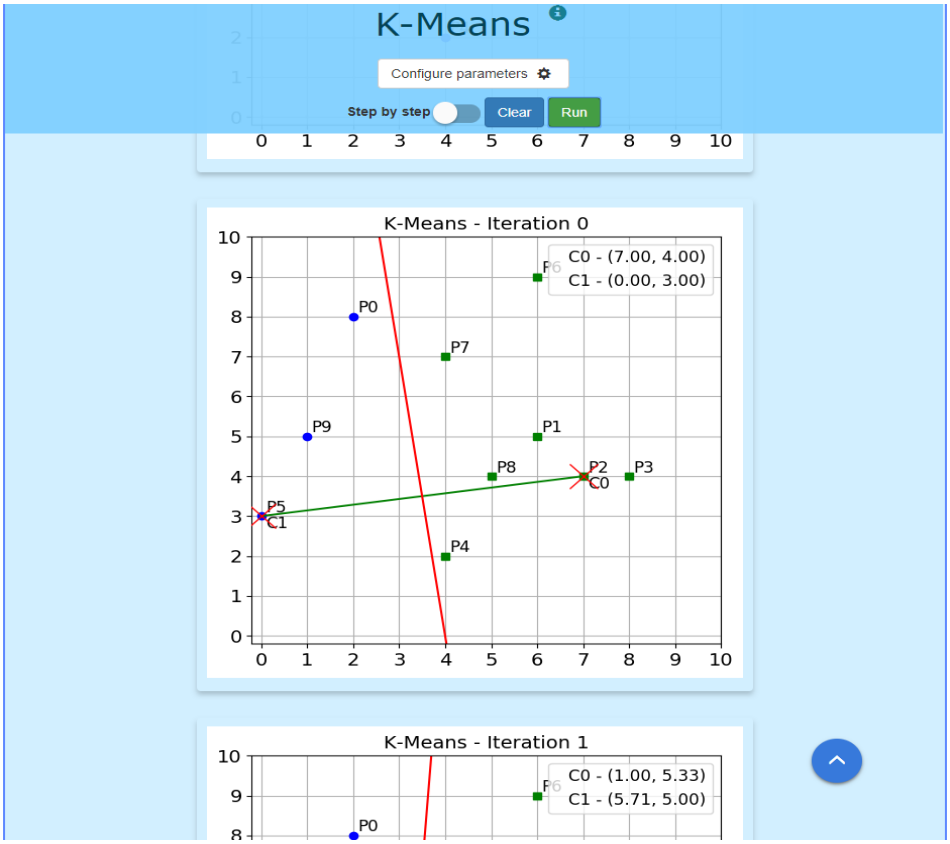


Figura 21. Schermata del sito con K-Means in esecuzione

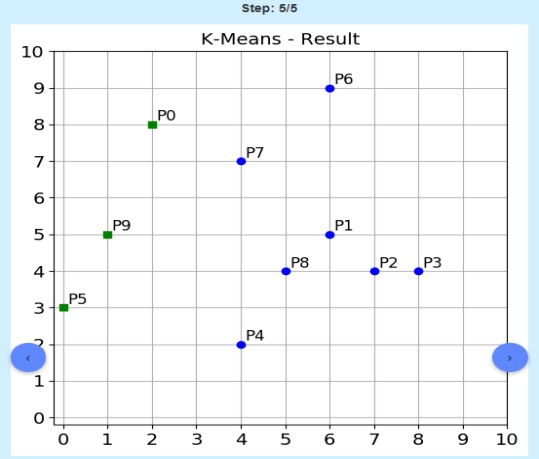


Figura 22. Schermata del sito con K-Means in esecuzione passo-passo

Nell'immagine sopra si vede K-Means in esecuzione passo-passo. Come prima indicazione il numero di iterazione corrente rispetto al totale (quinta iterazione su un totale di cinque), poi passando con il mouse sopra l'immagine appaiono dei piccoli pulsanti che permettono di passare all'iterazione precedente o a quella successiva.
Sotto sono presenti altre foto di altri algoritmi in esecuzione.

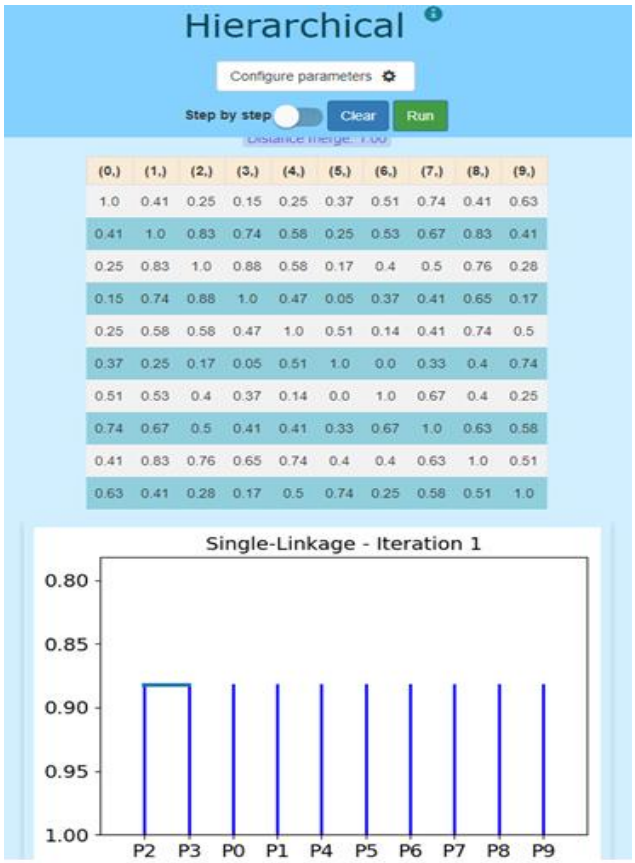


Figura 23. Schermata del sito con Hierarchical in esecuzione

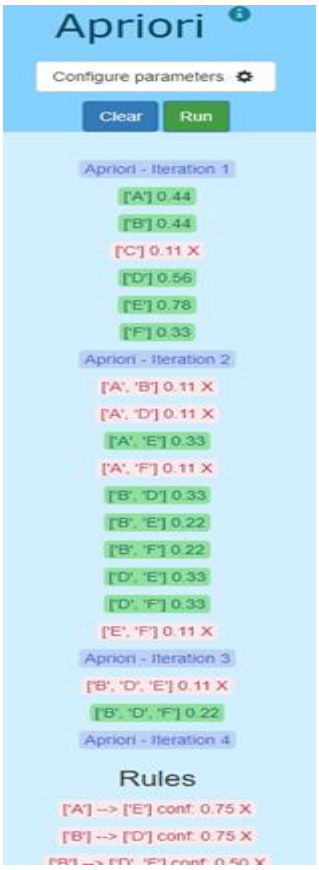


Figura 24. Schermata del sito con Apriori in esecuzione

Un inconveniente che può nascere in un layout cosi verticale può essere quello di dovere scorrere sempre verso la cima della pagina per andare a modificare tutte le opzioni d'esecuzione. È stato quindi introdotto un apposito pulsante con lo scopo di portare l'utente in cima alla pagina *(*Figura 25. Bottone per torna in cima alla pagina*)*.

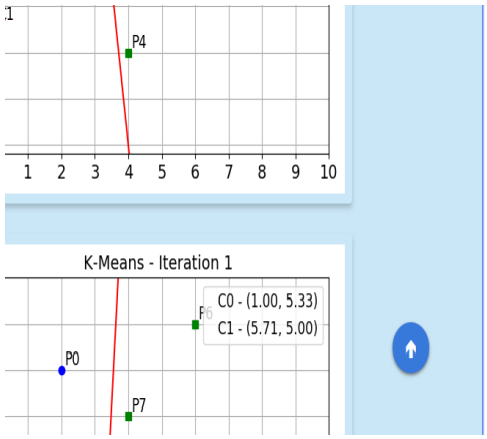


Figura 25. Bottone per torna in cima alla pagina

Come già accennato durante la fase introduttiva uno degli obiettivi del sito è quello di essere perfettamente usabile su ogni piattaforma, *smartphone* incluso. Ciò è stato reso possibile mediate un'accurata scelta di istruzioni CSS, sono quindi state fornite misure in percentuale e non assolute in modo tale da rendere agevole lo *scaling* da uno schermo a un altro. Un ruolo decisivo è stato giocato anche dal *framework* Bootstrap il quale centra il suo punto di forza sulla *responsiveness*, occupandosi quindi in automatico di ottimizzare al meglio lo spazio su schermo dei contenuti, laddove il risultato non fosse soddisfacente ho implementato le mie *media-query* ottenendo così la facoltà di personalizzare l'aspetto del sito sotto ogni punto di vista.
È possibile osservare tre immagini della *web application* in esecuzione su due dispositivi con schermo diverso (**Errore. L'origine riferimento non è stata trovata.**, **Errore. L'origine riferimento non è stata trovata.**, **Errore. L'origine riferimento non è stata trovata.**), immagini ottenute utilizzando l'emulatore di dispositivo integrato in Google Chrome.

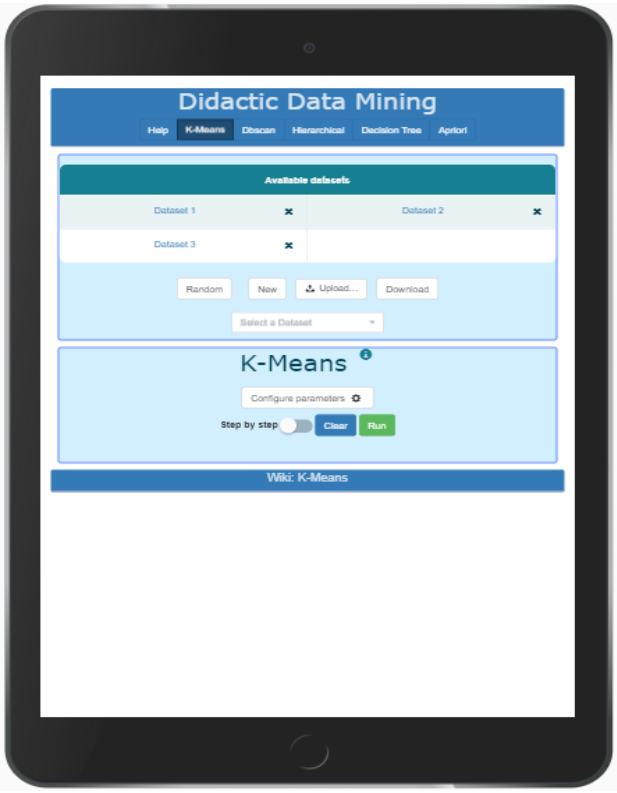


Figura 26. Web application in versione per tablet

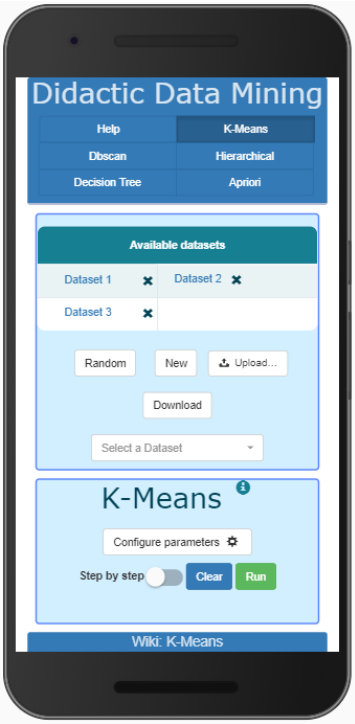


Figura 26. Web application in versione per smartphone

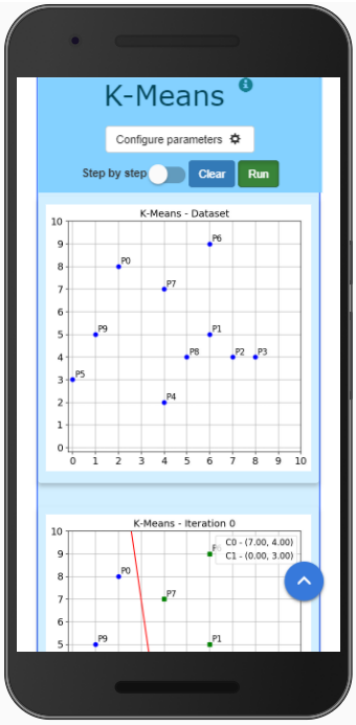


Figura 28. Web application in versione per smartphone in esecuzione

6 Conclusioni

Lo scopo del lavoro è stato quello di realizzare una *Web Application* finalizzata a aiutare gli studenti del corso di Laurea Magistrale in Informatica e Informatica Umanistica che seguono il corso di Data Mining con la finalità di facilitare la comprensione dei vari algoritmi studiati mediante l'analisi del loro comportamento al variare dei parametri e dei dati inseriti dagli utenti. Durante la redazione di questa relazione sono state prima introdotte nozioni di Data Mining e E-learning per poi passare alla discussione delle tecnologie utilizzate. Infine, è stato presentato il risultato ottenuto da un punto di vista progettuale e grafico, durante l'analisi è stato messo in evidenza il buon grado di scalabilità in base alla piattaforma usata e modularità della suddetta *web application,* caratteristiche che si riveleranno molto utili nel caso di eventuali espansioni future. Si ritiene quindi il prodotto completo in quanto viene offerto sia il supporto a tutti gli algoritmi presentati a lezione sia alle varie funzioni che un tipico utente si aspetta di avere a disposizione, quali creazione, modifica, cancellazione, caricamento e scaricamento di dataset. Da un punto di vista grafico il sito si presenta più *user-friendly* possibile cercando di mettere a proprio agio l'utente. La scelta degli elementi da visualizzare ridotti all'essenziale, l'uso di forme arrotondate, di colori che richiamano le tonalità del blu e di qualche piccola animazione, contribuiscono a creare un clima di semplicità e di armonia.

7 Bibliografia e sitografia

1. Pang-Ning Tan, Michigan State University, Michael Steinbach, University of Minnesota Vipin Kumar, *Introduction to Data Mining*, University of Minnesota,2006
2. Jennifer Niederst Robbins, *Learning Web Design: a beginners guide to HTML, CSS, Javascript and web graphics*, Quarta edizione, O'Reilly, 2012
3. Enrico Amedeo. *jQuery,* Apogeo, 2012
4. Guide e tutorial sulla programmazione web (https://www.w3schools.com), consultato 3/05/17
5. Introduzione a Bootstrap (https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-introduction.php), consultato 14/06/17
6. Guida sulla configurazione e installazione di Bootstrap (http://getbootstrap.com), consultato 15/06/17
7. V. Ambriola. *Programmazione in JavaScript*, consultato 12/12/16
8. Manuale di jQuery (https://api.jquery.com), consultato 8/05/17
9. Introduzione a Flask (http://flask.pocoo.org/docs/0.12/tutorial), consultato 8/05/17
10. Manuale di Flask (http://flask.pocoo.org/docs/0.12), consultato 25/06/17
11. Approfondimento su Kmeans (http://datamining.awardspace.com/samuele/kmenas.pdf), consultato 1/08/17
12. Decision Tree (https://en.wikipedia.org/wiki/Decision_tree_learning), consultato 11/08/17
13. E-Learning (https://community.articulate.com/series/getting-started/articles/what-is-e-learning), consultato 22/09/17
14. E-Learning (https://it.wikipedia.org/wiki/Apprendimento_online), consultato 22/09/17
15. Pagina web dello script usato per generare la pagina di caricamento (https://gasparesganga.com/labs/jquery-loading-overlay/), consultato 13/09/17
16. Manuale di Bootbox (http://bootboxjs.com/), consultato 17/06/17

8 Indice delle figure