

# Progetto Quality Treshold - Documentazione

A cura di: Vito Stefano Birardi

8 giugno 2025

## Indice

<b>1 Introduzione</b>	<b>4</b>
<b>2 Cos'è l'Algoritmo Quality Threshold</b>	<b>4</b>
<b>3 Principio di Funzionamento</b>	<b>4</b>
<b>4 Caratteristiche Distintive</b>	<b>4</b>
4.1 Vantaggi . . . . .	4
4.2 Svantaggi . . . . .	5
<b>5 Ambiti di Applicazione</b>	<b>5</b>
<b>6 Guida all'installazione</b>	<b>6</b>
6.1 Installazione JDK . . . . .	6
6.2 Variabili di ambiente . . . . .	7
6.3 Installazione MySQL . . . . .	7
6.4 Schermata iniziale . . . . .	7
6.5 Download dei file . . . . .	7
6.6 Configurazione del account MySQL . . . . .	8
6.7 Schermata finale . . . . .	9
<b>7 Eseguire il progetto base</b>	<b>10</b>
7.1 Importazione del database . . . . .	10
7.2 Normale funzionamento . . . . .	10
7.3 Struttura del progetto base . . . . .	11
<b>8 Avvio dell'applicazione</b>	<b>11</b>
8.1 Esecuzione di QTB_Server . . . . .	11
8.2 Esecuzione di QTB_Client . . . . .	12
<b>9 Avvio dall'ambiente di sviluppo</b>	<b>13</b>
<b>10 Test progetto base</b>	<b>14</b>
10.1 Gestione del raggio . . . . .	15
10.2 Uscita dal programma . . . . .	15
<b>11 Eseguire il progetto esteso</b>	<b>16</b>
11.1 Modifiche implementative . . . . .	16
11.2 Premessa . . . . .	16
<b>12 Eseguire l'applicazione</b>	<b>17</b>
12.1 Avvio dell'applicazione . . . . .	17
<b>13 Utilizzo del bot Telegram</b>	<b>17</b>
13.1 Schermata iniziale . . . . .	18
13.2 Comandi del bot . . . . .	18
13.3 Esecuzione dell'algoritmo QT . . . . .	19
13.4 Salvare i risultati . . . . .	20
13.5 Caricare un risultato precedente . . . . .	21
13.6 Ritorno al menù principale . . . . .	21

<b>14 Test progetto esteso</b>	<b>22</b>
14.1 Tabelle non trovate . . . . .	22
14.2 raggio non valido . . . . .	22
14.3 File non trovato . . . . .	23

## 1 Introduzione

Il progetto in questione utilizza l'algoritmo **Quality Threshold (QT)** per l'analisi dei dati. Tali dati possono essere estratti da un file o dalle tabelle di un database MySQL.

## 2 Cos'è l'Algoritmo Quality Threshold

L'algoritmo Quality Threshold è un algoritmo di **clustering** deterministico che appartiene alla famiglia degli algoritmi di raggruppamento gerarchico. A differenza degli algoritmi di partizionamento tradizionali, il QT non richiede di specificare a priori il numero di cluster da creare, ma si basa su un parametro di soglia di qualità (*threshold*) che definisce il raggio massimo consentito per ogni cluster.

## 3 Principio di Funzionamento

L'algoritmo si basa sulla **minimizzazione della distanza** tra i punti dati e il centroide del cluster di appartenenza. Il processo di clustering avviene attraverso i seguenti passaggi:

1. **Selezione del punto candidato:** Per ogni punto del dataset, viene valutata la possibilità di creare un nuovo cluster
2. **Formazione del cluster:** Vengono raggruppati tutti i punti che si trovano entro il raggio di soglia specificato
3. **Selezione del cluster ottimale:** Viene scelto il cluster che contiene il maggior numero di punti
4. **Iterazione:** Il processo si ripete sui punti rimanenti fino a quando tutti i punti sono stati assegnati a un cluster

## 4 Caratteristiche Distintive

L'algoritmo Quality Threshold presenta diverse caratteristiche che lo distinguono da altri approcci di clustering:

### 4.1 Vantaggi

- **Determinismo:** L'algoritmo restituisce sempre lo stesso risultato quando viene eseguito ripetutamente sullo stesso dataset, a differenza di algoritmi come il *k-means* che possono produrre risultati diversi a seconda dell'inizializzazione casuale
- **Numero di cluster automatico:** Non richiede di specificare il numero di cluster da creare, ma solo il raggio massimo (*threshold*), rendendo l'algoritmo più flessibile per dataset con strutture sconosciute
- **Robustezza al rumore:** È stato progettato per gestire efficacemente dati rumorosi e con distribuzioni non lineari

- **Approccio gerarchico:** Utilizza una strategia gerarchica per la creazione dei cluster, garantendo una maggiore stabilità dei risultati

## 4.2 Svantaggi

- **Complessità computazionale:** Richiede una potenza di calcolo maggiore rispetto ad algoritmi di partizionamento come il *k-means*, con una complessità temporale che può essere significativamente elevata per dataset di grandi dimensioni
- **Sensibilità al parametro threshold:** La scelta del raggio di soglia influenza notevolmente la qualità del clustering ottenuto

## 5 Ambiti di Applicazione

L'algoritmo Quality Threshold è particolarmente adatto per contesti in cui:

- I dati presentano rumore significativo
- La distribuzione dei dati è non lineare o presenta forme complesse
- Non si conosce a priori il numero ottimale di cluster
- È richiesta riproducibilità dei risultati
- La qualità del clustering è più importante dell'efficienza computazionale

## 6 Guida all'installazione

Prima di essere in grado di eseguire il programma, è necessario eseguire il file **risorse.bat** contenuto nella cartella Risorse.

Una volta eseguito il file, si aprirà una pagina di Powershell e seguirà un download.



Figura 1: Schermatta di download

Al termine del download del file compresso delle risorse, verranno estratti i file necessari per l'esecuzione del programma.

Per il progetto in questione è necessario installare il Java Developer Kit (JDK) nella versione 22.0.1 e il software di gestione del database MySQL nella sua versione 8.0.39.

La prima scheda di installazione che apparirà è quella del JDK.

Verrà chiesto se si vuole eseguire l'installazione del JDK tramite permessi di amministratore, cliccare su Sì.

### 6.1 Installazione JDK

Una volta confermata l'esecuzione come amministratore, si procederà all'installazione del JDK.



Cliccare su **Avanti**,nuovamente **Avanti** e infine, nel caso in cui l'installazione sia andata a buon fine, si dovrà cliccare sul tasto **Chiudi**.

## 6.2 Variabili di ambiente

è consigliato creare una variabile di ambiente per il JDK. Per farlo, si dovrà cercare la voce **variabili di ambiente** nella barra di ricerca di Windows e cliccare su Modifica le variabili di ambiente per il tuo account.

Successivamente si dovrà cliccare sul pulsante **Variabili di ambiente** e si aprirà una schermata con le variabili di ambiente.

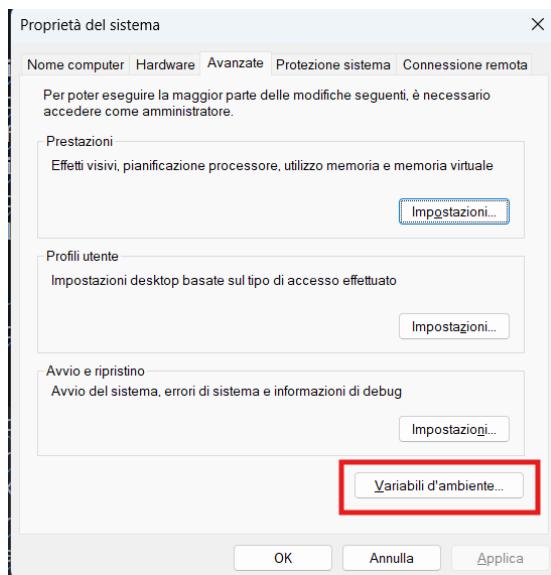


Figura 2: Schermata delle variabili di ambiente

Nella schermata, si dovrà cliccare sulla variabile PATH e modificarla in modo da aggiungere il percorso del JDK, inserendo il percorso in cui è stato installato il JDK.

Una volta completata l'installazione del JDK, si procederà automaticamente con l'installazione del MySQL.

## 6.3 Installazione MySQL

Anche qui sarà necessario fornire i permessi di amministratore, cliccare dunque su Sì.

## 6.4 Schermata iniziale

Dopodiché si aprirà la schermata di installazione di MySQL.

Selezionare il tipo di setup **Full** e cliccare su **Next**. Tale scelta permetterà di installare tutti i componenti necessari per l'esecuzione del programma.

## 6.5 Download dei file

Si passerà alla schermata di download dei file necessari per l'installazione, cliccare su **Execute**.

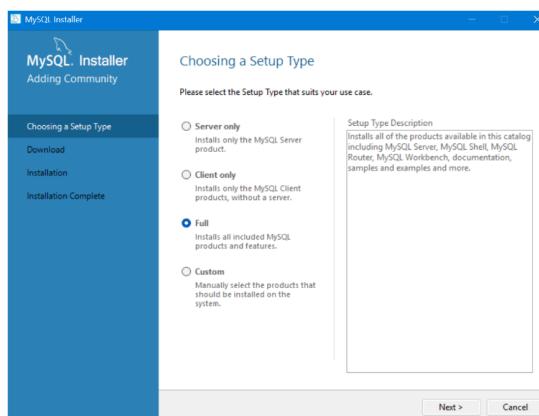


Figura 3: Scelta del setup

**Avvertenza**

Tale schermata potrebbe richiedere un po' di tempo per il download dei file, a seconda della velocità della connessione internet.

Dopo aver scaricato i file, si dovrà cliccare nuovamente su **Execute**.

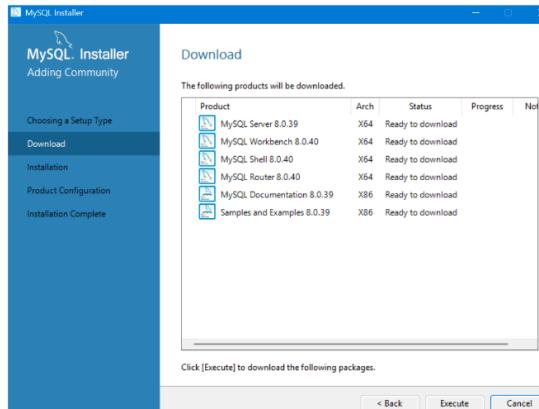


Figura 4: Download MySQL

## 6.6 Configurazione del account MySQL

Una volta terminato il download, si procederà con la configurazione dell'account MySQL. Si dovrà quindi scegliere una password, necessaria per accedere al database.

Si giungerà infine alla schermata di applicazione della configurazione del database. Cliccare su **Execute** per applicare la configurazione.

Una volta terminata anche l'installazione del MySQL, si dovrà cliccare su **Finish** per chiudere l'installler.

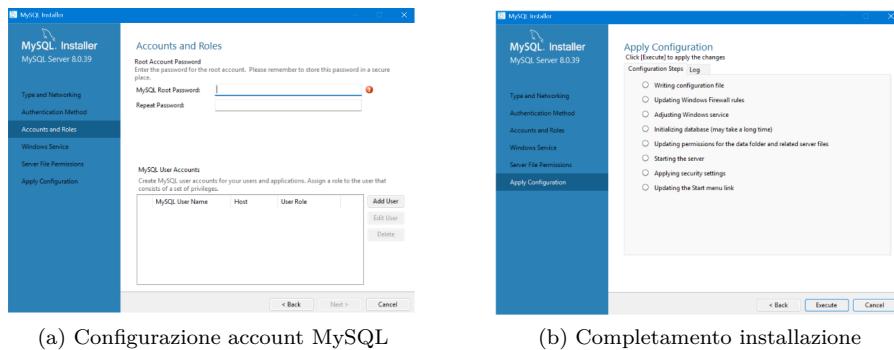


Figura 5: Configurazione e completamento MySQL



## 6.7 Schermata finale

Al termine dell'installazione il programma `resources.bat` mostrerà la seguente schermata.



Figura 6: Completamento installazione

## 7 Eseguire il progetto base

### 7.1 Importazione del database

Per il corretto funzionamento del progetto base, è necessario, prima di eseguire il programma, importare il database `QT.sql` presente nella cartella `Progetto_MAP_2/risorse/` all'interno del proprio DBMS.

Siccome nella sezione precedente abbiamo fornito l'installer per Mysql, si consiglia di utilizzare quest'ultimo per importare il database. Per importare il database all'interno di Mysql, è necessario aprire il terminale, accedere con le credenziali di root a Mysql, e lanciare il seguente comando:

```
source *PERCORSO DEL FILE QT.sql*
```

#### Avvertenza

Si ricorda di cambiare il percorso del file `QT.sql` con quello corretto, effettuando un *drag and drop* del file all'interno del terminale, in modo da evitare errori di battitura.

Se il comando è stato eseguito correttamente, il terminale restituirà la seguente schermata:

### 7.2 Normale funzionamento

La base dell'algoritmo permette all'utente di scegliere un raggio massimo per il cluster.

Per ciascun punto interessato, ossia i punti con distanza inferiore al raggio impostato, l'algoritmo calcola i cluster candidati.

Una volta individuati i cluster, quello con il numero maggiore di punti verrà salvato nella cartella `./results/`, ossia nella root del progetto base. Il nome del file indicherà il database di partenza e il raggio massimo utilizzato per la ricerca, quindi scegliendo per esempio il database `gopicnic` con un raggio di 3, il file di output sarà `gopicnic_003.dmp`.

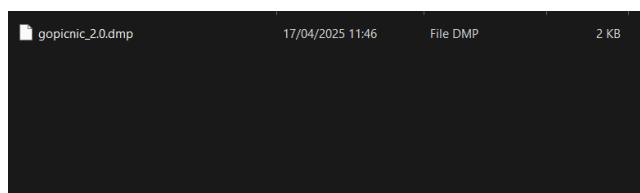


Figura 7: Risultati del salvataggio

Una volta creato il file di output, l'algoritmo provvederà ad eliminare anche i cluster non salvati, per evitare che vengano esaminati nuovamente.

Nel caso in cui più di un cluster abbia il numero massimo di punti, l'algoritmo ripeterà la procedura con il set ridotto di punti.

### 7.3 Struttura del progetto base

Il progetto base consiste in un'applicazione di tipo client/server. Di fatto, nella cartella **Jar+Bat** sono presenti i file **start\_server.bat** per l'esecuzione del server e **start\_client** per il client, che non sono altro che degli script batch per l'esecuzione dei file jar.

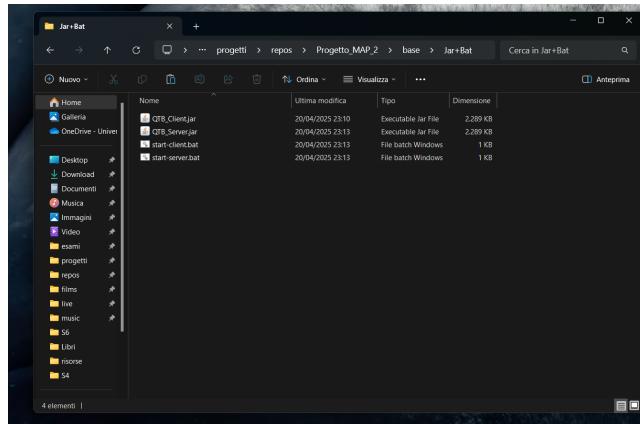


Figura 8: Jar del progetto base

#### Avvertenza

Per avviare il programma correttamente, è necessario eseguire prima il file **.bat** del server e poi quello del client.

## 8 Avvio dell'applicazione

### 8.1 Esecuzione di QTB\_Server

Avviando il file **start\_server.bat** verrà mostrata la seguente schermata, che indica che il server è in ascolto sulla porta 8080, in attesa di una richiesta da parte del client.

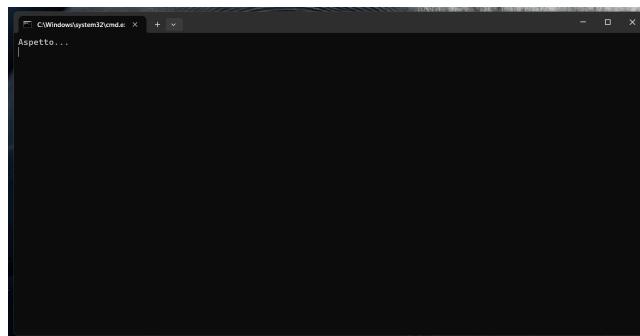


Figura 9: Server in ascolto

## 8.2 Esecuzione di QTB\_Client

Per quanto riguarda il client, una volta avviato, mostrerà la seguente schermata.

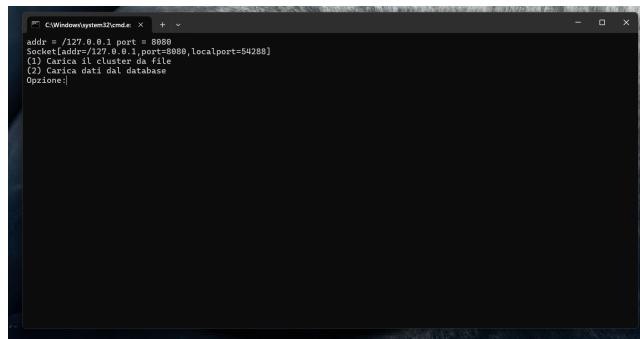
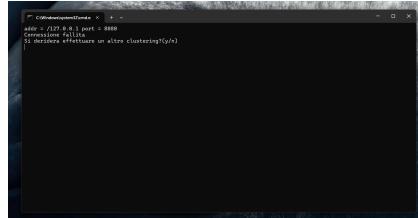


Figura 10: Schermata iniziale del client

### Avvertenza

Nel caso in cui il server non si sia avviato correttamente, o sia stato avviato il client prima del server, il client mostrerà la seguente schermata, che indicherà appunto che la connessione al server non è avvenuta correttamente.



Una volta assicurati che il client comunichi correttamente con il server, si potrà decidere se caricare i dati da un file di un clustering precedente o da un database.

Se si sceglie di caricare i dati da una tabella del database, si procederà con l'inserimento del nome della tabella da cui si vogliono estrarre i dati. Nel caso in cui tale tabella non sia presente nel database, il client avviserà l'utente con un messaggio di errore e chiedendo di inserire un nome valido.

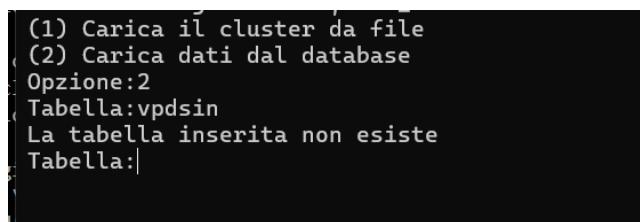


Figura 11: Errore tabella non presente

Una volta scelto correttamente una tabella, si procederà nel determinare il raggio del clustering e, automaticamente, il programma restituirà il risultato.

```

Tabella:gopicnic
Raggio:2
Numero di cluster5
1:Centroid=(rain 13.0 high weak no )
Examples:
[rain 13.0 high weak no ] dist=0.0
AvgDistance=0.0

2:Centroid=(sunny 0.1 normal weak no )
Examples:
[sunny 0.1 normal weak no ] dist=0.0
[sunny 40.0 low weak no ] dist=1.9975
AvgDistance=0.99875

3:Centroid=(sunny 30.3 high strong no )
Examples:
[sunny 30.3 high strong no ] dist=0.0
[overcast 12.5 high strong no ] dist=1.445
AvgDistance=0.7225

4:Centroid=(overcast 30.0 high weak yes )
Examples:
[sunny 30.3 high weak yes ] dist=1.0075
[overcast 30.0 high weak yes ] dist=0.0
[sunny 20.0 high weak yes ] dist=1.25
[overcast 29.21 normal weak yes ] dist=1.01975
[overcast 23.0 low weak yes ] dist=1.175
AvgDistance=0.8904500000000001

5:Centroid=(rain 0.0 normal strong no )
Examples:
[rain 40.0 high strong no ] dist=2.0
[rain 0.0 normal weak no ] dist=1.0
[rain 0.0 normal strong no ] dist=0.0
[overcast 0.1 normal strong no ] dist=1.0025
[rain 18.0 normal weak no ] dist=1.45
[sunny 12.5 normal strong no ] dist=1.3125
[rain 12.5 high strong no ] dist=1.3125
AvgDistance=1.1539285714285714

```

Figura 12: Risultato del clustering

A fine esecuzione, il risultato del clustering verrà salvato nella cartella `./results/` con il nome della tabella e il raggio utilizzato per il clustering, come già spiegato in precedenza.

## 9 Avvio dall'ambiente di sviluppo

È chiaramente possibile eseguire il progetto base senza passare per i file `.bat` messi a disposizione, ma direttamente da un IDE come Eclipse o IntelliJ.

In questo caso, sono necessarie alcune accortezze aggiuntive.

### Avvertenza

In questa sezione si fa riferimento all'utilizzo di Visual Studio Code, ma è possibile utilizzare qualsiasi IDE che supporti Java e abbia la possibilità di eseguire file jar.

Prima di tutto, occorre aprire la cartella del progetto base all'interno dell'IDE, in modo da avere un albero delle cartelle come quello mostrato in figura.

Una volta aspettato che Visual Studio Code abbia caricato il progetto (potrebbero volerci diversi minuti), si potrà procedere con l'importazione della libreria

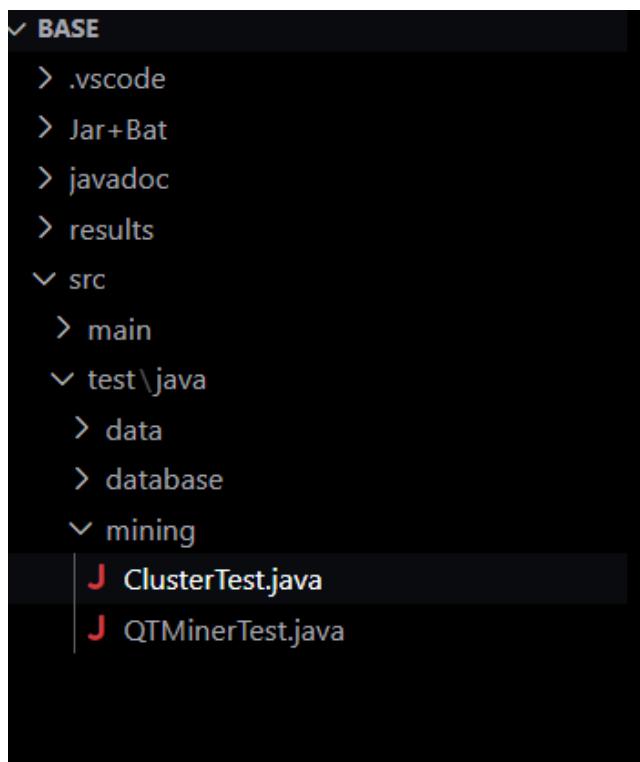


Figura 13: Sourcetree del progetto base

`mysql-connector-java-8.0.32.jar` all'interno del progetto, in modo da poter utilizzare le classi necessarie per la connessione al database.

Tale libreria è presente nella cartella `Progetto_MAP_2/risorse/` e può essere importata all'interno del progetto in Visual Studio Code seguendo questi passaggi:

1. cliccare sulla sezione `Java Projects` in basso a sinistra
2. trovare la cartella `Referenced Libraries`
3. cliccare con il tasto destro su `Referenced Libraries` e selezionare il tasto `+` per aggiungere una libreria esterna
4. selezionare il file `mysql-connector-java-8.0.32.jar` presente nella cartella `/Risorse/`
5. attendere che Visual Studio Code abbia caricato la libreria

## 10 Test progetto base

Durante l'esecuzione del programma, può capitare che l'utente inserisca dei valori che quest'ultimo non è in grado di gestire. Chiaramente nonostante la presenza di queste situazioni, deve essere garantita la continuità di esecuzione del programma e la corretta amministrazione di queste situazioni estreme.

### 10.1 Gestione del raggio

Nel caso in cui l'utente inserisca un valore del raggio maggiore del massimo raggio possibile, il programma restituirà un messaggio di errore e chiederà nuovamente il raggio.

Nel caso invece in cui l'utente inserisca un valore non numerico, il programma restituirà un messaggio di errore e chiederà nuovamente il raggio.

In entrambi i casi, il programma non si fermerà e continuerà a chiedere il raggio fino a quando non verrà inserito un valore corretto.

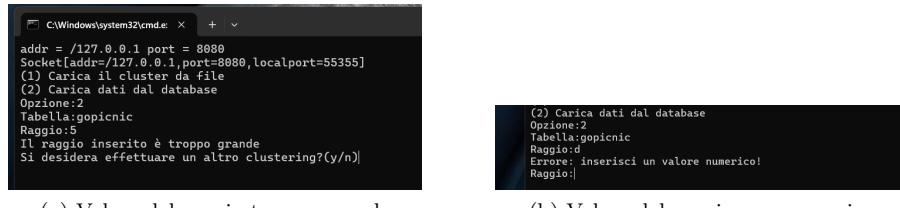


Figura 14: Errori nel valore del raggio

### 10.2 Uscita dal programma

Una volta calcolato il raggio, il programma chiederà all'utente se ha intenzione di effettuare nuovamente un clustering: in caso positivo chiederà nuovamente la misura del raggio, in caso opposto chiederà all'utente se vuole tornare al menù principale o chiudere il programma.



Figura 15: Gestione finale clustering

Domandando all'utente se si vuole ritornare al menù principale, si gestisce la chiusura del programma e, allo stesso tempo, il ritorno al menù principale, ossia alla situazione iniziale.

Inoltre, tale stratagemma previene lo spiacevole inconveniente per cui si è obbligati a forzare l'interruzione dell'esecuzione.

## 11 Eseguire il progetto esteso

### 11.1 Modifiche implementative

Rispetto alla versione base, la funzione del client è stata completamente stravolta. Nel progetto base, infatti, il client non era altro che un'estensione del server, che si occupava di inviare richieste ad esso e di stampare le risposte.

Ora, invece, tale funzione è stata delegata direttamente ad un bot telegram, il quale si occupa, come un vero e proprio client, di inviare le richieste al server e di stampare le risposte.

### 11.2 Premessa

Rispetto al progetto base, si è reso necessario l'utilizzo di un gestore di dipendenze, in modo da poter installare le librerie necessarie per il progetto.

#### Maven

Il gestore di dipendenze scelto per questo scopo è **Maven**, il quale permette di scaricare le librerie necessarie e di gestire le versioni delle stesse.

Il vantaggio principale rispetto ad altri gestori di dipendenze, come il più blasfomato Gradle, è la sua semplicità di utilizzo e la sua integrazione con gli IDE più diffusi.

Inoltre, al contrario di Gradle, Maven non richiede la creazione di un file di configurazione complesso, ma si basa su un semplice file **pom.xml** che contiene le informazioni sulle dipendenze del progetto.

Non richiede quindi la creazione di file di build o di scaricare i file delle dipendenze manualmente, ma il tutto viene gestito automaticamente.

#### Telegram API

Si è reso necessario l'utilizzo di un gestore delle dipendenze per via dell'utilizzo delle librerie **org.telegram.longpolling** e **org.telegram.telegrambots**, le quali permettono di interagire con l'API di Telegram.

## 12 Eseguire l'applicazione

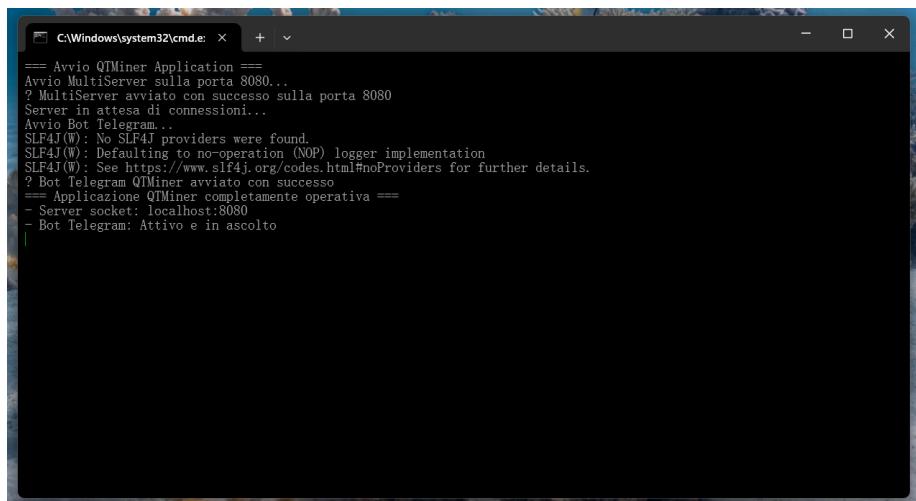
Rispetto alla versione base, l'usabilità del programma è migliorata. Se prima occorreva eseguire sia il server che il client, ora è possibile eseguire un solo file JAR, che corrisponde sia al server che al modulo di comunicazione con il bot Telegram.

### 12.1 Avvio dell'applicazione

Per avviare l'applicazione, come del resto nel progetto base, occorre spostarsi nella cartella **Jar+Bat** e eseguire il file **main.bat**.

A differenza del progetto bas , tuttavia, all'interno della cartella in questione sarà presente solo un file **.bat**. Pertanto, sarà sufficiente eseguire tale file per avviare l'applicazione, effettuando un doppio click su di esso.

Se il file **main.bat** si apre correttamente, verrà aperta una finestra del terminale, che mostrerà i seguenti logs:



```
== Avvio QTMiner Application ==
Avvio MultiServer sulla porta 8080...
? MultiServer avviato con successo sulla porta 8080
Server in attesa di connessioni...
Avvio Bot Telegram...
SLF4J(W): No SLF4J providers were found.
SLF4J(W): Defaulthing to no-operation (NOP) logger implementation
SLF4J(W): See https://www.slf4j.org/codes.html#noProviders for further details.
? Bot Telegram QTMiner avviato con successo
== Applicazione QTMiner completamente operativa ==
- Server socket: localhost:8080
- Bot Telegram: Attivo e in ascolto
```

Figura 16: Corretto avvio dell'applicazione

#### Avvertenza

Si raccomanda di essere connessi ad internet, in quanto il bot Telegram necessita di una connessione per poter funzionare correttamente. Nel caso in cui si avvisasse il jar del progetto esteso senza una connessione ad internet, il bot Telegram non funzionerà correttamente e non risponderà ai comandi inviati.

## 13 Utilizzo del bot Telegram

Una volta avviata la comunicazione tra il server e Telegram, sarà possibile utilizzare il bot Telegram per interagire con il server.

Tale bot è raggiungibile digitando il nome `@QualityThresholdBot` nella barra di ricerca di Telegram, oppure cliccando sul seguente link: [https://t.me/QT\\_Quality\\_Thresholdbot](https://t.me/QT_Quality_Thresholdbot)

### 13.1 Schermata iniziale

Per far avviare il bot, è sufficiente premere sul pulsante `/start` presente in basso, o in alternativa, digitare il comando `/start` nella barra di testo del bot. Questo step risulta necessario per avviare la comunicazione tra il bot e il server, in modo da poter inviare e ricevere comandi.



Figura 17: Schermata di avvio bot

Una volta premuto il tasto `/start`, il bot risponderà con un messaggio di benvenuto, in cui verrà spiegato brevemente il suo scopo e l'operazione da effettuare per iniziare ad utilizzarlo.

### 13.2 Comandi del bot

Il bot Telegram implementa i seguenti comandi, nella seguente disposizione:

- **Mostra tabelle:** mostra le tabelle dei risultati ottenuti con l'algoritmo QT.
- **Esegui Algoritmo QT:** permette di eseguire l'algoritmo QT sui dati caricati.
- **Carica Risultati:** permette di caricare i risultati ottenuti dall'algoritmo QT.
- **Carica Risultati:** permette di caricare i risultati ottenuti dall'algoritmo QT.
- **Salva Risultati :** permette di salvare i risultati ottenuti dall'algoritmo QT.
- **Torna al menù principale :** permette di tornare al menù principale del bot.

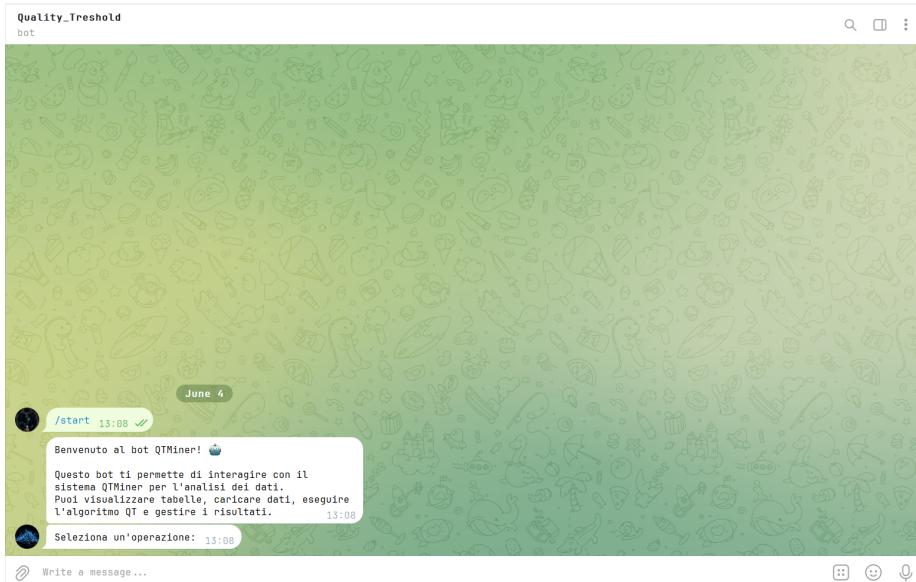


Figura 18: Messaggio di benvenuto del bot

### 13.3 Esecuzione dell'algoritmo QT

Chiaramente l'opzione più importante è quella che permette l'esecuzione di calcolo dell'algoritmo QT. Per eseguire l'algoritmo, è sufficiente premere il pulsante **Esegui Algoritmo QT**.

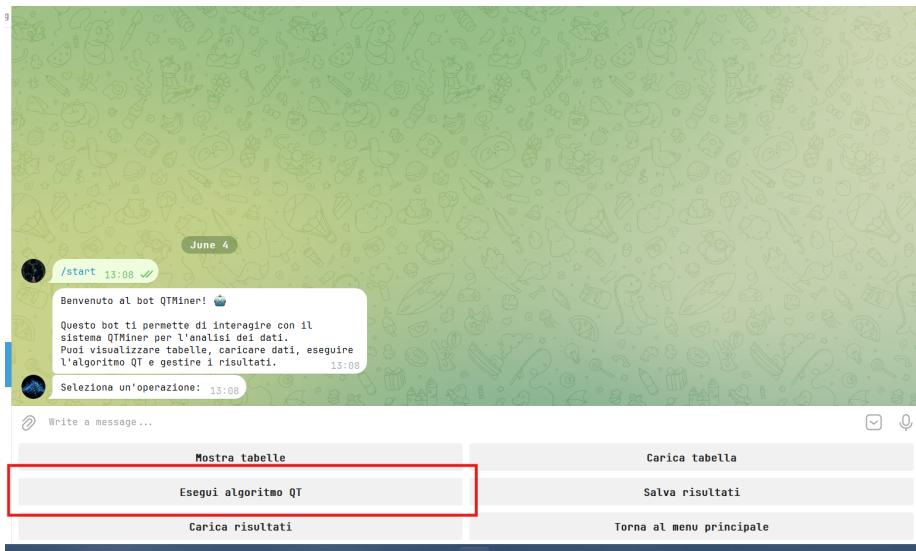


Figura 19: Opzione di esecuzione algoritmo

Una volta premuto il pulsante, il bot chiederà di inserire il raggio. Tuttavia, se prima di premerlo non è stato caricata alcuna tabella proveniente dal database,

il bot chiederà di caricarne una, in modo da poter eseguire l'algoritmo QT sui dati caricati.

Quindi, l'utente dovrà premere il pulsante **Carica Tabella** per caricare una tabella, e successivamente premere il pulsante **Esegui Algoritmo QT** per eseguire l'algoritmo QT, specificando gli opportuni parametri richiesti.



Figura 20: Processo di esecuzione dell'algoritmo QT

### 13.4 Salvare i risultati

Una volta eseguito l'algoritmo QT, il bot permetterà di salvare il risultato dell'esecuzione in un file **.dmp**, in maniera molto simile a come avveniva nel progetto base, all'interno della cartella **results**, presente nella cartella principale del progetto esteso.

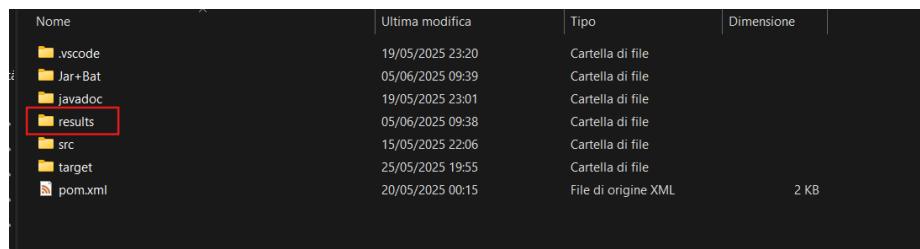


Figura 21: Posizione cartella results

Chiaramente per eseguire tale operazione, è necessario premere il pulsante **Salva Risultati**.

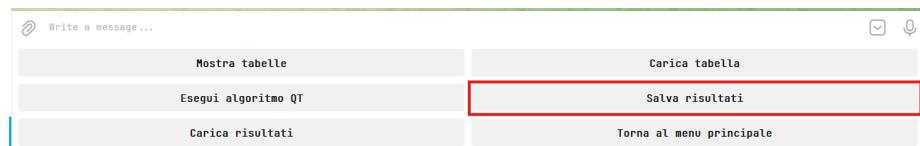


Figura 22: Opzione di salvataggio dei risultati

### 13.5 Caricare un risultato precedente

Qualora si voglia caricare un risultato precedentemente salvato, è possibile farlo premendo il pulsante **Carica Risultati**.



Figura 23: Opzione di caricamento dei risultati

Il bot non chiederà direttamente di inserire il nome del file da caricare ma, dato che il nome del file **.dmp** è costruito in modo da contenere le informazioni relative al raggio e alla tabella utilizzata per l'esecuzione dell'algoritmo QT, il bot chiederà di inserire il raggio e la tabella da ritrovare.

Se il file **.dmp** esiste, il bot risponderà restituendo il risultato dell'esecuzione in questione.



Figura 24: Risultato del caricamento di un file **.dmp**

### 13.6 Ritorno al menù principale

Qualora si voglia far ripartire il bot dalla schermata iniziale, è possibile farlo premendo il pulsante **Torna al menù principale**.

Una volta premuto il pulsante, il bot risponderà con un messaggio di benvenuto, come quello mostrato alla fine della sezione [13.1](#).

## 14 Test progetto esteso

Durante la fase di test del progetto esteso, sono stati eseguiti diversi test per verificare il corretto funzionamento delle varie funzionalità dei comandi implementanti. Tali test hanno coinvolto principalmente la correttezza della gestione delle varie situazioni anomale che possono incorrere durante l'esecuzione dei comandi, come ad esempio la gestione di file non trovati, errori di sintassi o problemi di accesso ai file.

Chiaramente queste situazioni anomale devono essere notificate al livello di client, per permettere all'utente di comprendere l'entità del problema e prendere le opportune misure.

Si fa notare che i test elencati sono stati eseguiti tenendo conto il normale flusso di lavoro del bot, nel senso che è stato seguito il flusso di interazione del bot per via sequenziale, senza saltare passaggi o inviare comandi in modo casuale.

### 14.1 Tabelle non trovate

Nel caso in cui venga inserito il nome di una tabella inesistente, il bot risponderà specificando che la tabella inserita non esiste e invita a inserire un nome di tabella valido.



Figura 25: comportamento tabella inesistente

Chiaramente l'algoritmo QT non potrà essere eseguito su una tabella inesistente, quindi il bot risponderà con un messaggio di errore specificando che non è possibile eseguire l'algoritmo QT se prima non viene caricata una tabella valida.



Figura 26: inconsistenza algoritmo QT

### 14.2 raggio non valido

Nel caso si inserisca un raggio non valido, che include quindi le casistiche in cui il raggio è negativo o non numerico, il bot risponderà specificando che il raggio inserito non è valido, in quanto genererebbe un solo cluster, e pertanto invita a inserire un raggio valido.

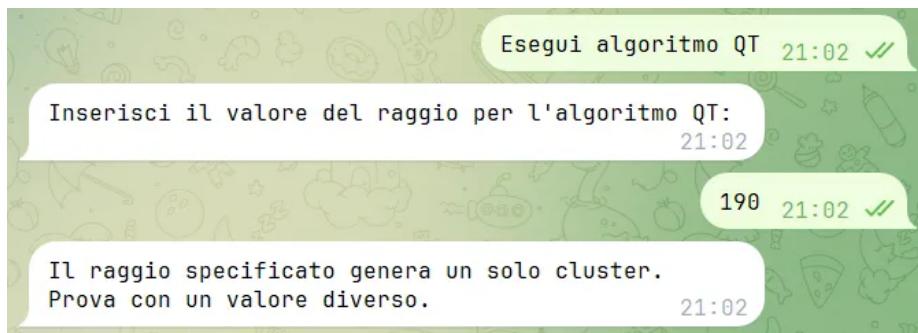


Figura 27: Raggio invalido

### 14.3 File non trovato

Nel caso in cui, durante l'esecuzione del comando `Carica risultati`, il file specificato non dovesse essere presente enlla cartella `results`, il bot risponderà specificando che il file non è stato trovato.

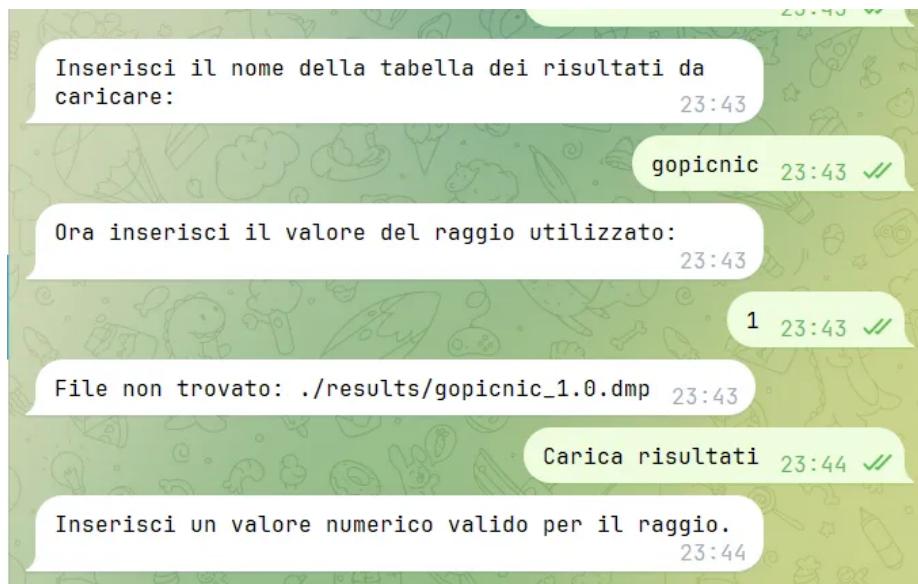


Figura 28: Esempio di file non trovato

Sarà quindi necessario eseguire nuovamente il comando ed inserire il nome di un file valido.