System Architecture and Implementation

In this section, we introduce the architecture and implementation of our ALMSS as shown in Fig. [1](#bookmark4). Visualization module is the core module of our system. Users can select the appropriate dataset for operation and input the SQL statement of the query. After learning the index, according to the results of the automatic selection of the model, users can view the accurate query location and the model information involved in the query process. In the Upload Dataset module, after the user clicks on a dataset, the corresponding dataset information will be displayed at the same time, including data set, data distribution, etc. In the SQL Parsing module, users enter the SQL statements to be queried. Our system does a regular parsing of this statement and gives a wide range of key position descriptions of the related fields. This description will be transmitted to the Learning Index module for further processing. This section provides some models and algorithms for users to reduce the range of keys to the greatest extent. Automatic Selection module receives the results of the previous module. It automatically selects the appropriate regression model for analysis to get the accurate query location, and provides the selected model and its parameters to users.

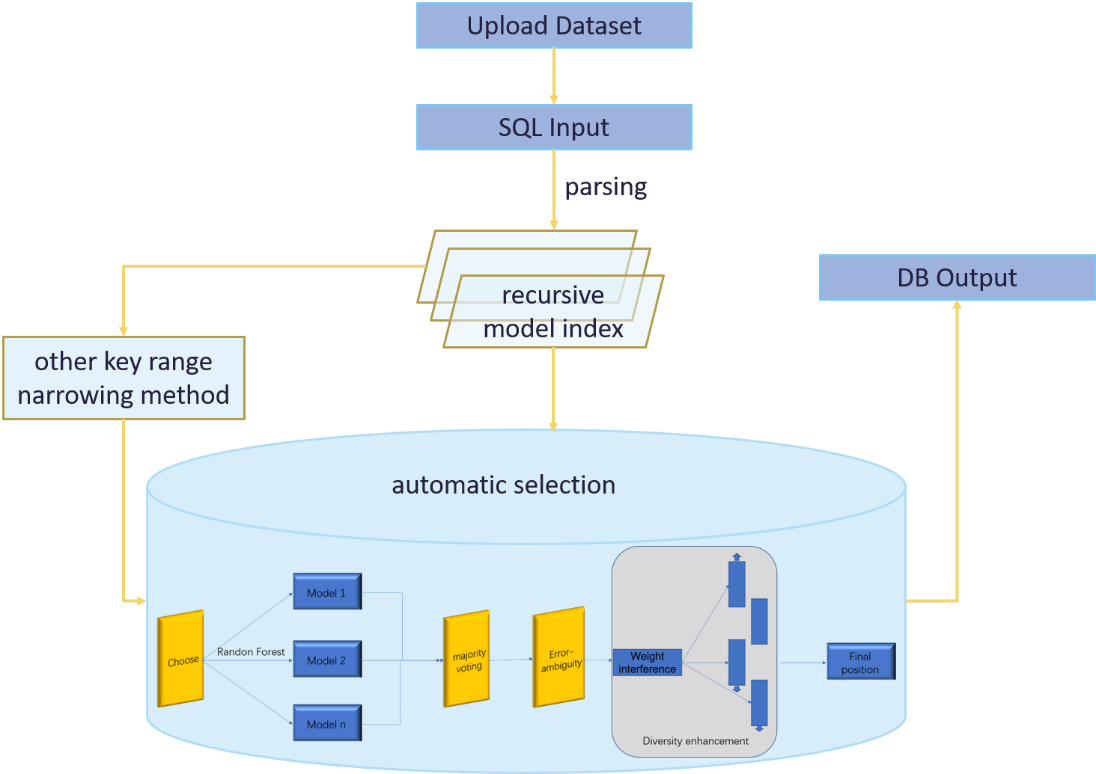


Fig. 1 System Architecture and implementation of ADSML

Considering that Python integrates many machine learning libraries which helps to build our model, our system is mainly constructed in Python and C++。And we choose the well-known MySQL as the storage basement.

1. Demonstration Scenario

We plan to demonstrate our ALMSS from the following four parts.

**—** *Upload Dataset.* As shown in Fig. [2a,](#bookmark12) we provide a visual operation button and information display interface. Users can view the existing dataset and select the one they need according to its content, data distribution and other information.

**—***Sql Parsing.* As depicted in Fig. [2b,](#bookmark11) users enter the SQL statement that needs to be queried, and ALMSS does regular parsing. We associate the SQL statement with the corresponding fields and calculate a wide range of keys ready to be processed in the next stage. In this part, we also present a schematic diagram of the entire process.

**—***Learning Index.* As Fig. [2c](#bookmark13) presents, the ALMSS system provides three schemes for rapidly reducing the range of keys. We presents the recursive model index (RMI), traditional Btree and Hash, so that users can view the visual build process time and other information of the different schemes we presented.

**—***Automatic Selection.* As Fig. [2d](#bookmark13) presents, ALMSS automatically selects the appropriate regression model for training according to the minimum key range. Users can view the selected optimal model, relevant parameter information, training evaluation result, and the exact location of the query.