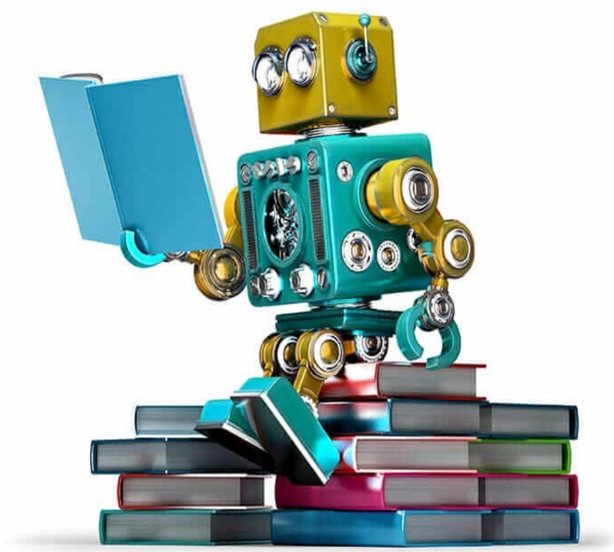
‘Hyper’Parameters—predicted right!

Manasa Vanga, Tejas Munot
vanga.m@husky.neu.edu, munot.t@husky.neu.edu

INFO 6210, Spring 2019, Northeastern University



Our very own Machine Learning Superhero!

1. Introduction

In the good old days, computers had to be programmed and told what to do. In some cases, you even had to tell them how to do it. Life was monotonous back then. But Arthur Samuel didn't like it. He saw that there's more to it, and thus began the chapter of Machine Learning!

Machine learning is the art of teaching the machine how to learn. It can thus, identify patterns and make decisions without humans telling it to do so.

How does it do it? you ask. The answer is by building analytical models. A machine learning model is a mathematical formula having some parameters which are needed to be learned from the data through model training. Thus, we fit the model parameters. But some parameters cannot be learned directly through model training. These are called **Hyperparameters**.

Hyperparameters express the properties of the model. For example, its complexity, how fast should the model learn. These must be specified **before** the model training process starts. Thus, knowledge of the hyperparameters is important as it has a large effect on the predictive power of the statistical model and that is what we, students at Northeastern University, Boston are working on under the AI SkunkWorks association.

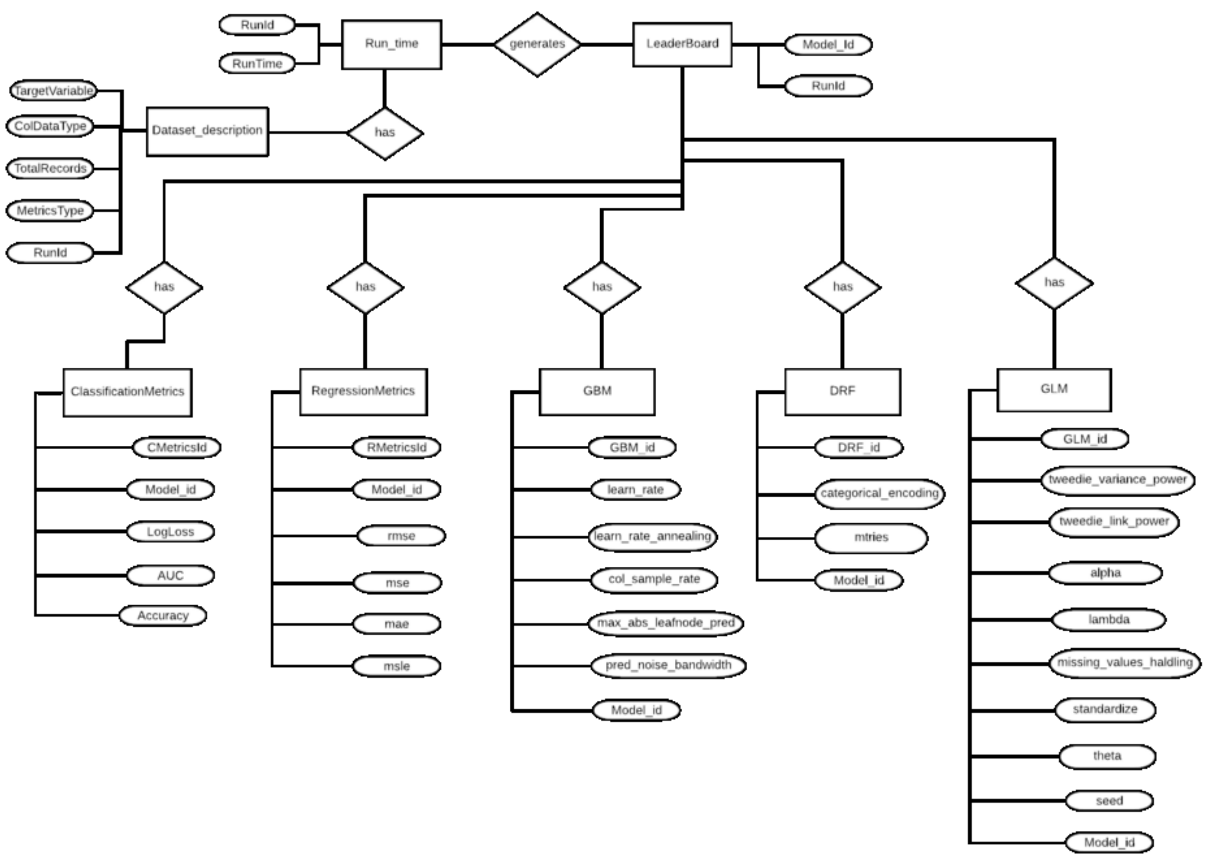
Our aim is to create a physical database which would support a website where data enthusiasts could upload their datasets and based on the millions of dataset records we have, give the user the best values of the respective hyperparameters for whatever he/she plans to achieve through the model. With this, we can help the data scientists save a lot of their time which they can utilize to find that one missing semicolon in the hundreds of lines of code! Fun isn't it?

We are a team of 4 students, 2 aspiring data scientists, Prakruthi and Urja, and 2 database engineers, myself and Manasa. The data science (DS) team plans to use H2O which is a fully open source, distributed in-memory machine learning platform with linear scalability. H2O supports the most widely used statistical & machine learning algorithms including gradient boosted machines, generalized linear models, deep learning and more. H2O also has an industry leading **AutoML** functionality that automatically runs through all the algorithms and their hyperparameters to produce a leaderboard of the best models.

2. Code with Documentation

2.1 Conceptual Diagram

We then identified the entities, attributes, and relationships. Thus, we came up with the first version of our conceptual diagram.

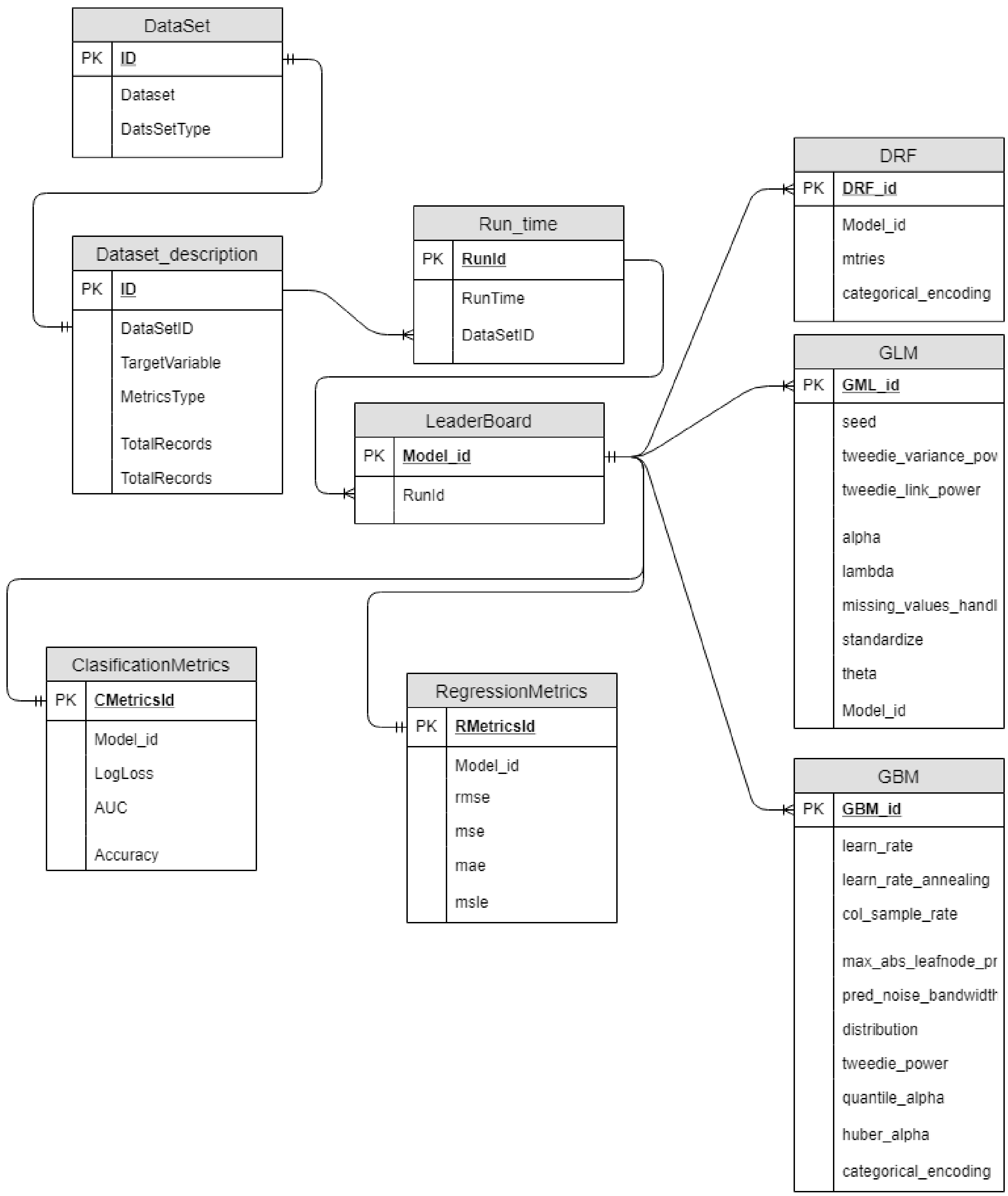


Conceptual Diagram—Iteration 1

But, as it always does, our conceptual diagram evolved throughout the course of the project. As we went ahead and addressed the uses cases for the database, we had to change the conceptual diagram

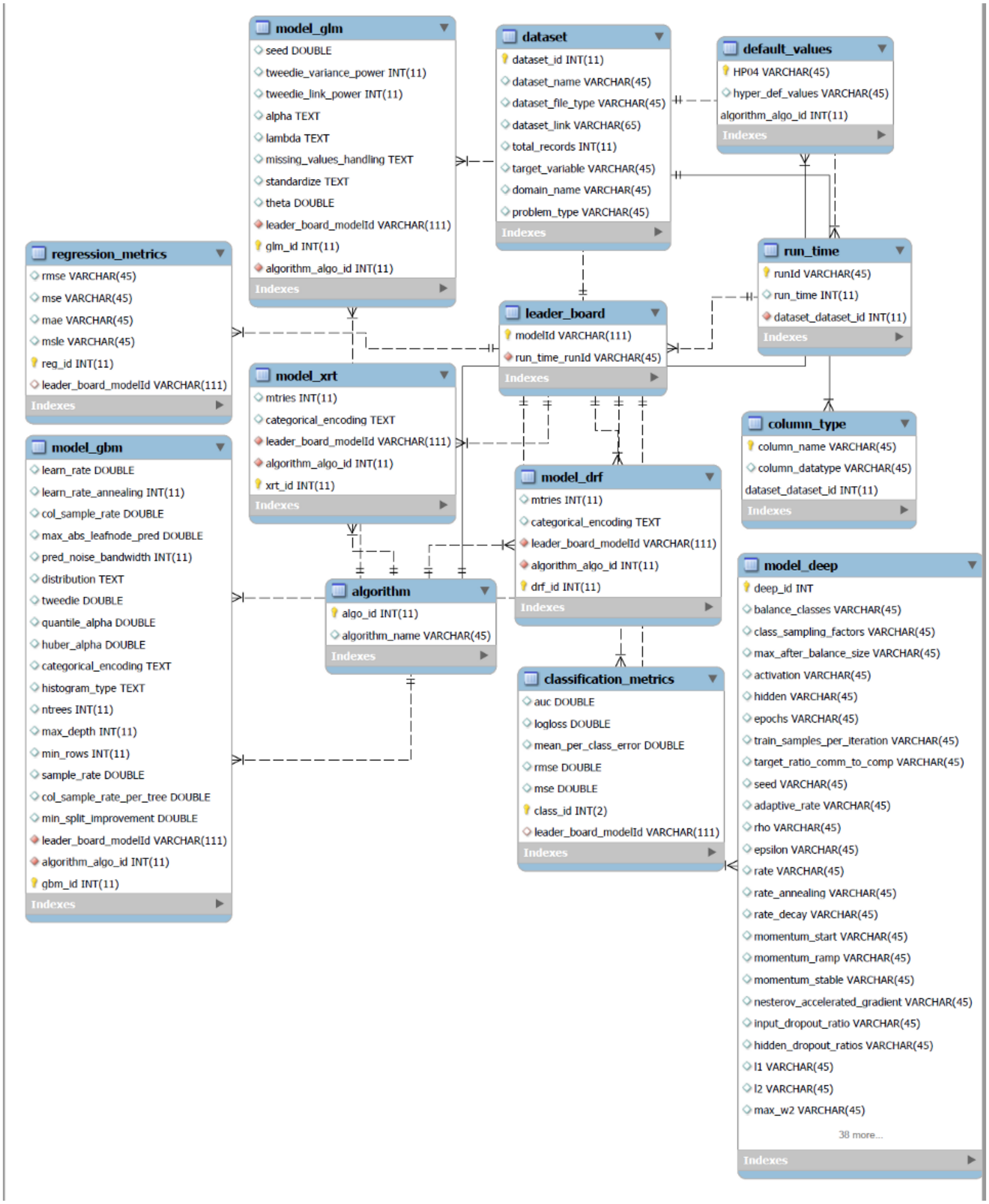
Based on the conceptual diagram, we made the Entity-Relationship Diagram (ERD) addressing the cardinalities. We were now ready to create the physical database. We used MySQL Workbench extensively. All our queries are written and executed in MySQL Workbench.

2.2 ER—Diagram



Revised ER –Diagram:

The Final ERD is as follows:



Entity Relationship Diagram

2.3.Tables

- **Entity Sets:**

The Final tables and their description

1. **Dataset**—It has all the information about the dataset. (PK—dataset_id)
2. **Run_time**—It has the runtimes and the run-Id's for each runtime (PK—runId; FK—dataset_dataset_id)
3. **Leader_board**—It has the model-Id's each run has generated (PK—modelId; FK—run_time_runId)
4. **Default_values**—It has all the hyperparameters for each algorithm and its default values (PK—HP04)
5. **Column_type**—It has the dataset's information specific to the columns and its data types given to H20. (PK—column_name, dataset_dataset_id)
6. **Classification_metrics**—It has the classification metrics for each classification dataset (PK—classId; FK—leader_board_modelId)
7. **Regression_metrics**—It has the regression metrics for each regression dataset (PK—reg_id; FK—leader_board_modelId)
8. **Algorithm**—All the algorithms stored in the table (PK—algo_id)
9. **Model_glm**—All the models from the GLM algorithm which got generated during running H20 (PK—glm_id; FK—algorithm_algo_id, leader_board_model)
10. **Model_gbm**—All the models from the GBM algorithm which got generated during running H20(PK—gbm_id; FK—algorithm_algo_id, leader_board_model)
11. **Model_xrt**—All the models from the XRT algorithm which got generated during running H20(PK—xrt_id; FK—algorithm_algo_id, leader_board_model)

12. **Model_deep**—All the models from the Deep Learning algorithm which got generated during running H20(PK—deep_id; FK—algorithm_algo_id, leader_board_model)

13. **Model_drf**—All the models from the DRF algorithm which got generated during running H20(PK—drf_id; FK—algorithm_algo_id, leader_board_model)

- **RelationShips:**

  1. R1 – dataset_id, runId
     One dataset can run for multiple runtimes. This is a one to many relationship between dataset and run_time.

  2. R2 - modelId, run_time_runId
     For one run time ,multiple models from different algorithms are generated. It is a one to many relationship between leader_board and run_time.

  3. R3 - PK—glm_id; FK—(algorithm_algo_id, leader_board_model)
     For one Algorithm and leader_board_model, there are multiple models has its hyperparameters. It is a one to many relationship between model_gbm and leader_board.

     Similar relationship exists for other Algorithms and is models.

- **Normalization:**

1. First normal form (1NF):

- Each table has a primary key
- The values in each column of a table are atomic. No multi-value attributes found.
- No repeating groups in all the tables.

2. Second normal form (2NF):

- All requirements for 1st NF are met.
- Partial dependencies are eliminated by creating separate entity sets.

- There is no calculated data

3. Third normal form (3NF):

- All requirements for 2nd NF are met.
- Fields that do not directly depend on the primary key are removed and creates as separate entities thus removed transitive dependencies.

2.4 Creating the physical database

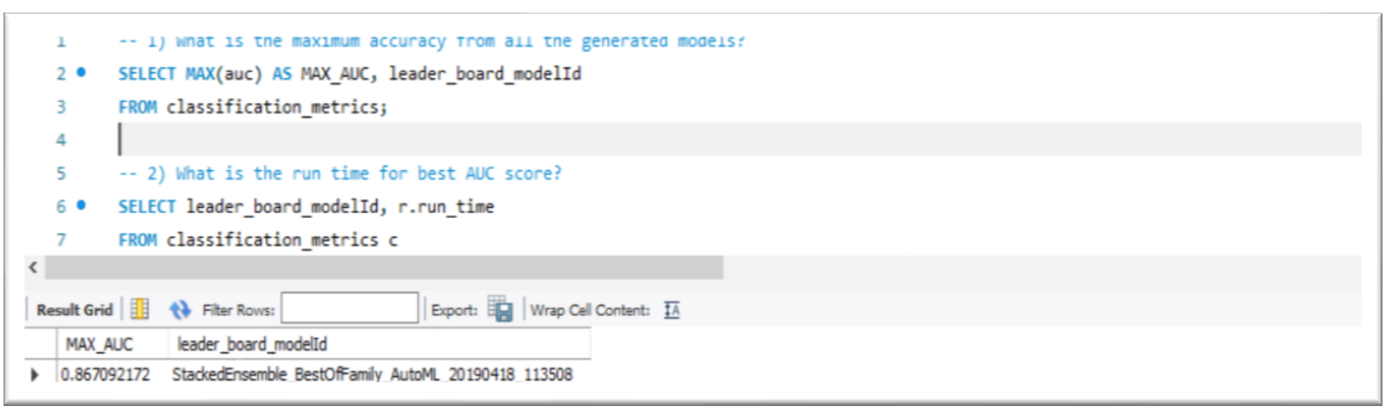
After we finalized the ERD, we imported data into the tables. Then, using Forward Engineering, we got all the tables along with the foreign constraints and thus our schema was ready for querying.

3.Use Cases:

We then got together with the Data Science (DS) team to come up with some practical queries which could help the data scientists.

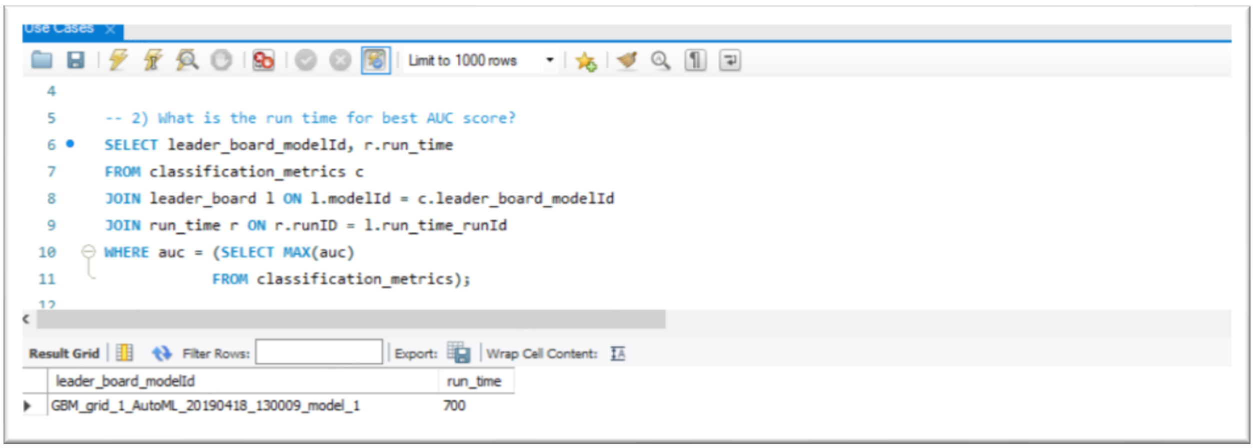
1. What is the maximum accuracy from all the generated models?

SELECT MAX(auc) AS MAX_AUC, leader_board_modelId
FROM classification_metrics;



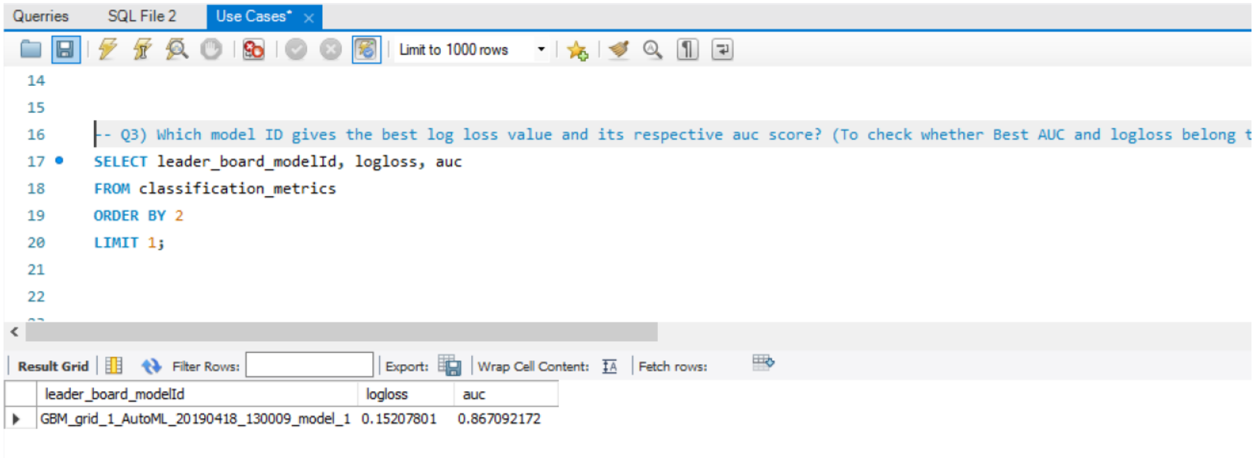
2) What is the run time for the best AUC score?

SELECT leader_board_modelId, r.run_time
FROM classification_metrics c
JOIN leader_board l ON l.modelId = c.leader_board_modelId
JOIN run_time r ON r.runID = l.run_time_runId
WHERE auc = (SELECT MAX(auc)
FROM classification_metrics);



Q3) Which model ID gives the best log loss value? (To check whether Best AUC and logloss belong to the same model?)

SELECT leader_board_modelId, logloss, auc
FROM classification_metrics
ORDER BY 2
LIMIT 1;

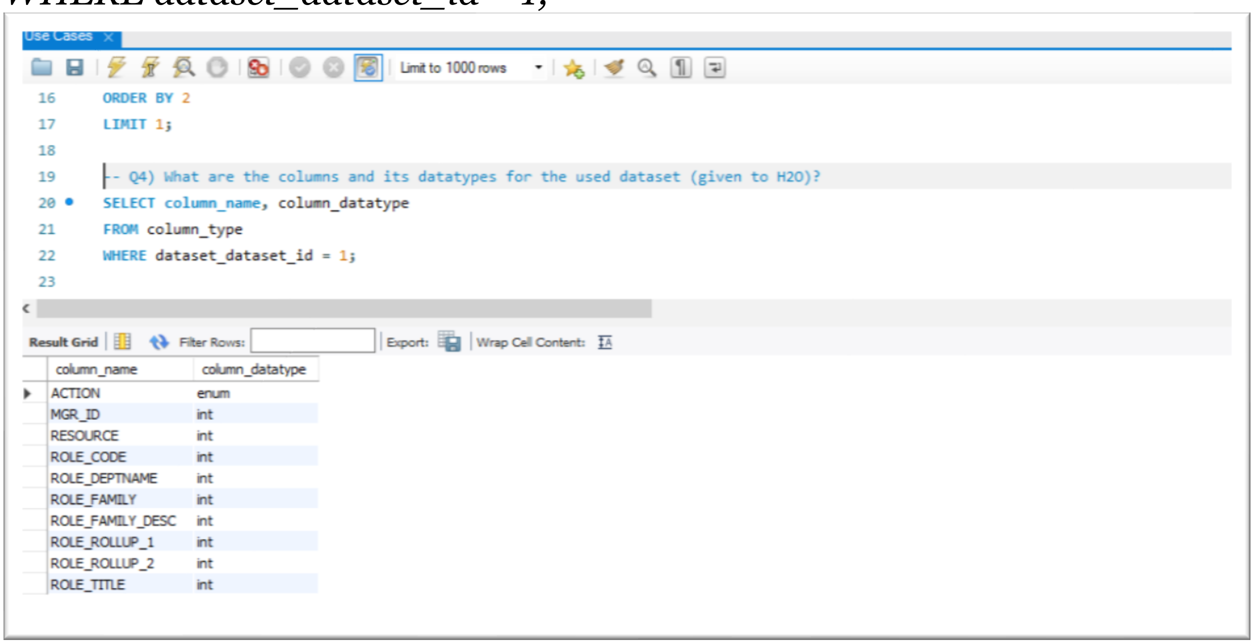



```
-- Q3) Which model ID gives the best log loss value and its respective auc score? (To check whether Best AUC and logloss belong t
SELECT leader_board_modelId, logloss, auc
FROM classification_metrics
ORDER BY 2
LIMIT 1;
```

leader_board_modelId	logloss	auc
GBM_grid_1_AutoML_20190418_130009_model_1	0.15207801	0.867092172

Q4) What are the columns and its datatypes for the used dataset (given to H2O)?

SELECT column_name, column_datatype
FROM column_type
WHERE dataset_dataset_id = 1;



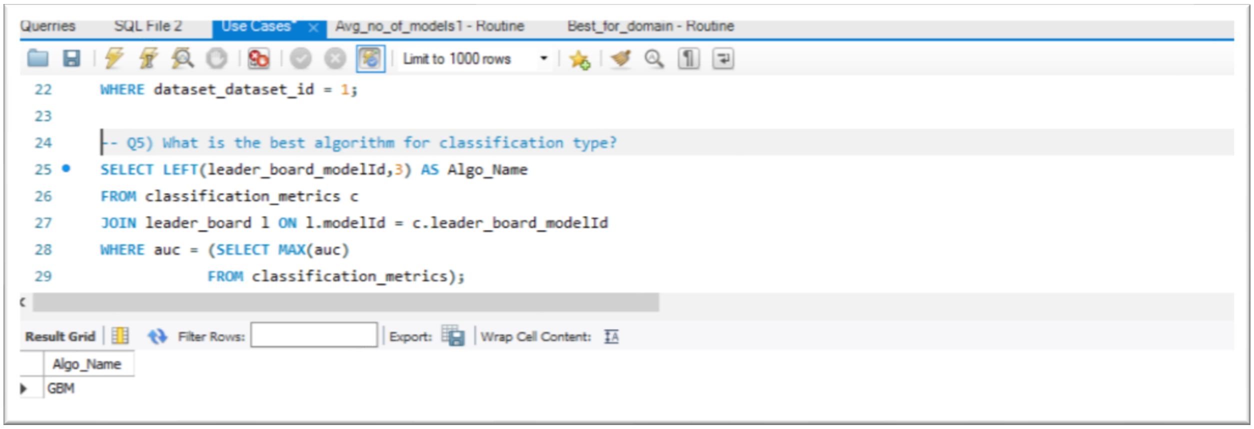

```
ORDER BY 2
LIMIT 1;

-- Q4) What are the columns and its datatypes for the used dataset (given to H2O)?
SELECT column_name, column_datatype
FROM column_type
WHERE dataset_dataset_id = 1;
```

column_name	column_datatype
ACTION	enum
MGR_ID	int
RESOURCE	int
ROLE_CODE	int
ROLE_DEPTNAME	int
ROLE_FAMILY	int
ROLE_FAMILY_DESC	int
ROLE_ROLLUP_1	int
ROLE_ROLLUP_2	int
ROLE_TITLE	int

Q5) What is the best algorithm for classification type?

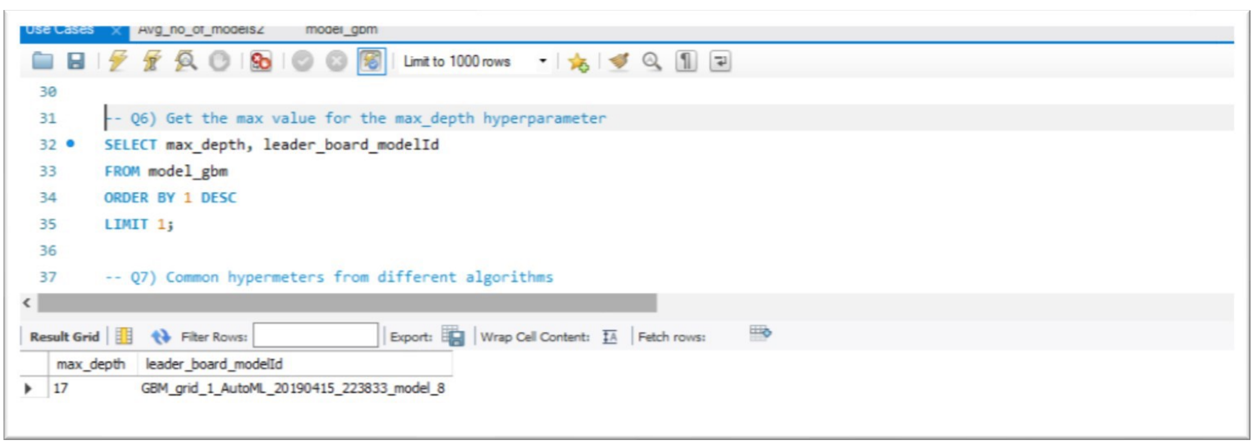
SELECT LEFT(leader_board_modelId,3) AS Algo_Name
FROM classification_metrics c

JOIN leader_board l ON l.modelId = c.leader_board_modelId
WHERE auc = (SELECT MAX(auc)
FROM classification_metrics);



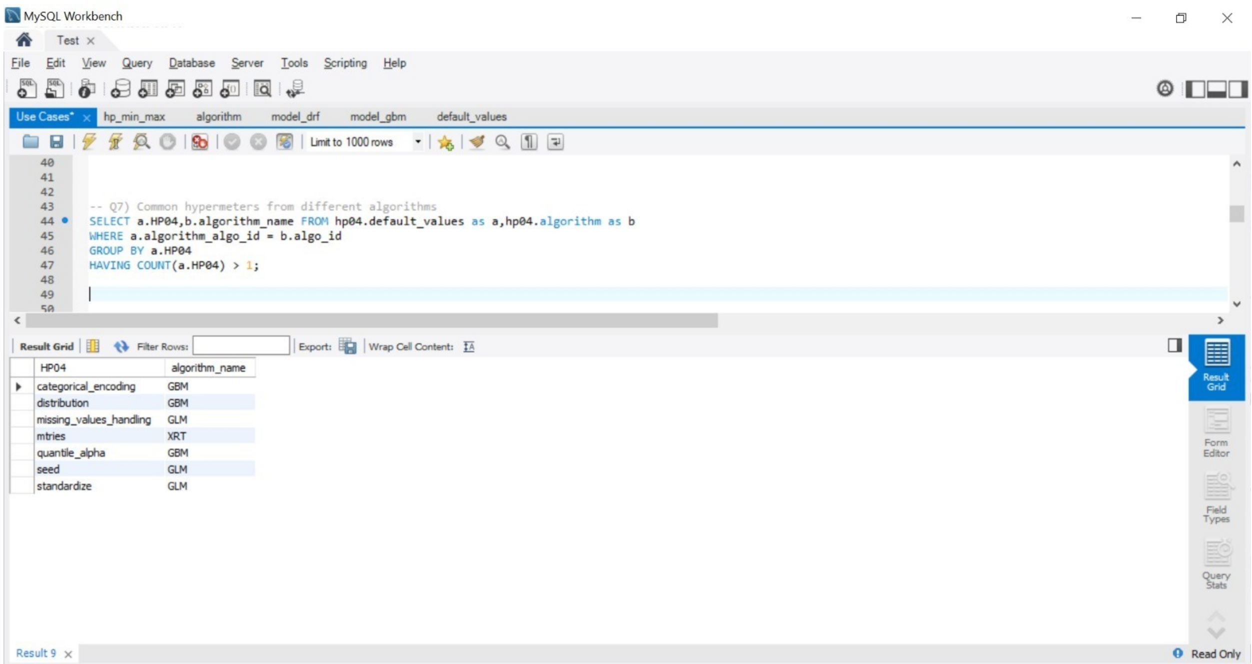
Q6) Get the Max value of the max_depth hyperparameter

SELECT max_depth, leader_board_modelId
FROM model_gbm
ORDER BY 1 DESC
LIMIT 1;



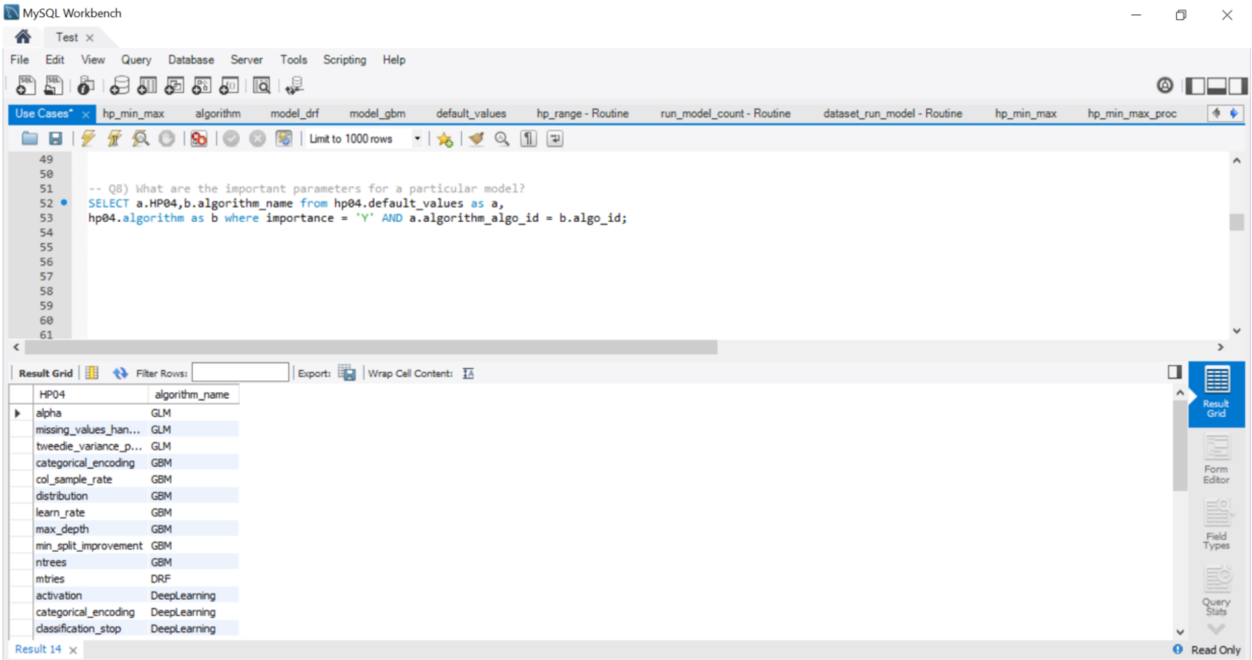
Q7) Common hypermeters from different algorithms

SELECT a.HP04,b.algorithm_name FROM hp04.default_values as a,hp04.algorithm as b
WHERE a.algorithm_algo_id = b.algo_id
GROUP BY a.HP04
HAVING COUNT(a.HP04) > 1;

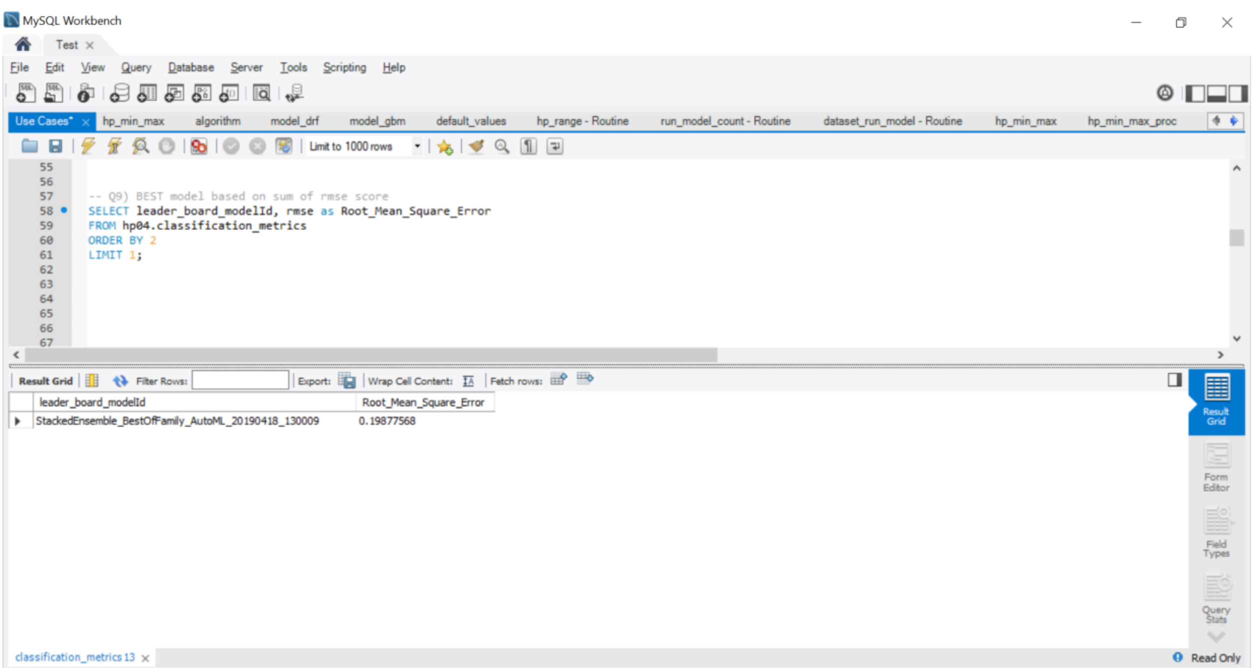


Q8) What are the important parameters for a particular model?

SELECT a.HP04,b.algorithm_name from hp04.default_values as a, hp04.algorithm as b where importance = 'Y' AND a.algorithm_algo_id = b.algo_id;

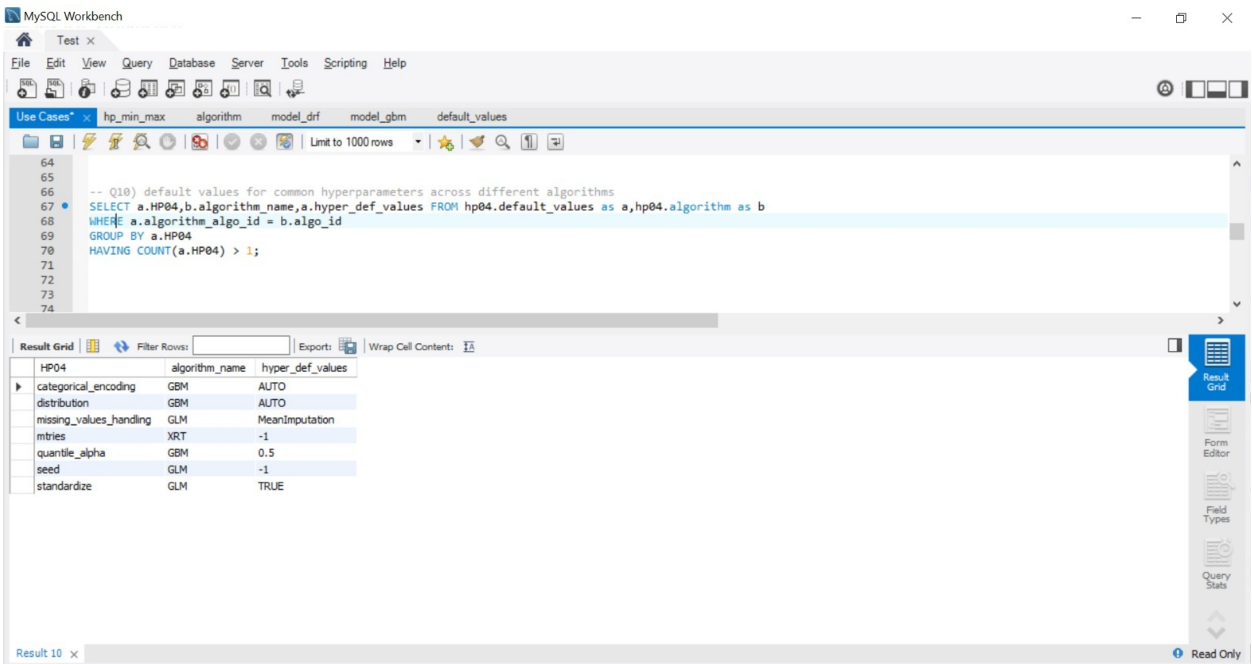


Q9) BEST model based on rmse score:
SELECT leader_board_modelId, rmse as Root_Mean_Square_Error
FROM hp04.classification_metrics
ORDER BY 2
LIMIT 1;



Q10) Default values for common hyperparameters across different algorithms

SELECT a.HP04,b.algorithm_name,a.hyper_def_values FROM hp04.default_values as a,hp04.algorithm as b
WHERE a.algorithm_algo_id = b.algo_id
GROUP BY a.HP04
HAVING COUNT(a.HP04) > 1;



4. Functions and stored procedures:

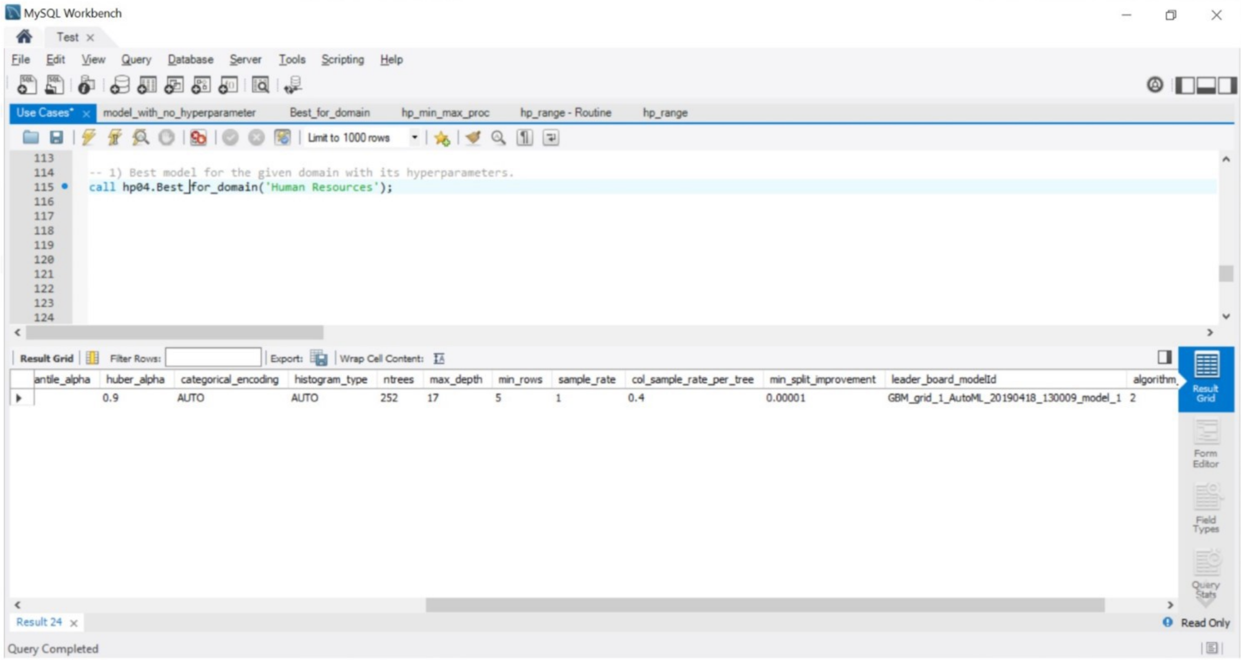
1. The best model for the given domain with its hyperparameters.

Input: Domain Name

Output: The best model with its actual hyperparameter values

CODE:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`Best_for_domain`(IN domain_name1 VARCHAR(45))
BEGIN
DECLARE model VARCHAR(111);
SELECT c.leader_board_modelId INTO model
FROM classification_metrics c
JOIN leader_board l ON l.modelId = c.leader_board_modelId
JOIN run_time r ON r.runID = l.run_time_runId
JOIN dataset d ON r.dataset_dataset_id = d.dataset_id
WHERE domain_name = domain_name1
AND auc = (SELECT MAX(auc)
FROM classification_metrics);
IF(LEFT(model,3) = 'GBM')
THEN
SELECT * from model_gbm WHERE leader_board_modelId = model;
ELSEIF(LEFT(model,3) = 'GLM') THEN
SELECT * from model_glm WHERE leader_board_modelId = model;
ELSEIF(LEFT(model,3) = 'DRF') THEN
SELECT * from model_drf WHERE leader_board_modelId = model;
ELSEIF(LEFT(model,3) = 'XRT') THEN
SELECT * from model_xrt WHERE leader_board_modelId = model;
END IF;
END$$
DELIMITER ;
```



2. What are the minimum and maximum values of learn_rate, ntrees hyperparameter?

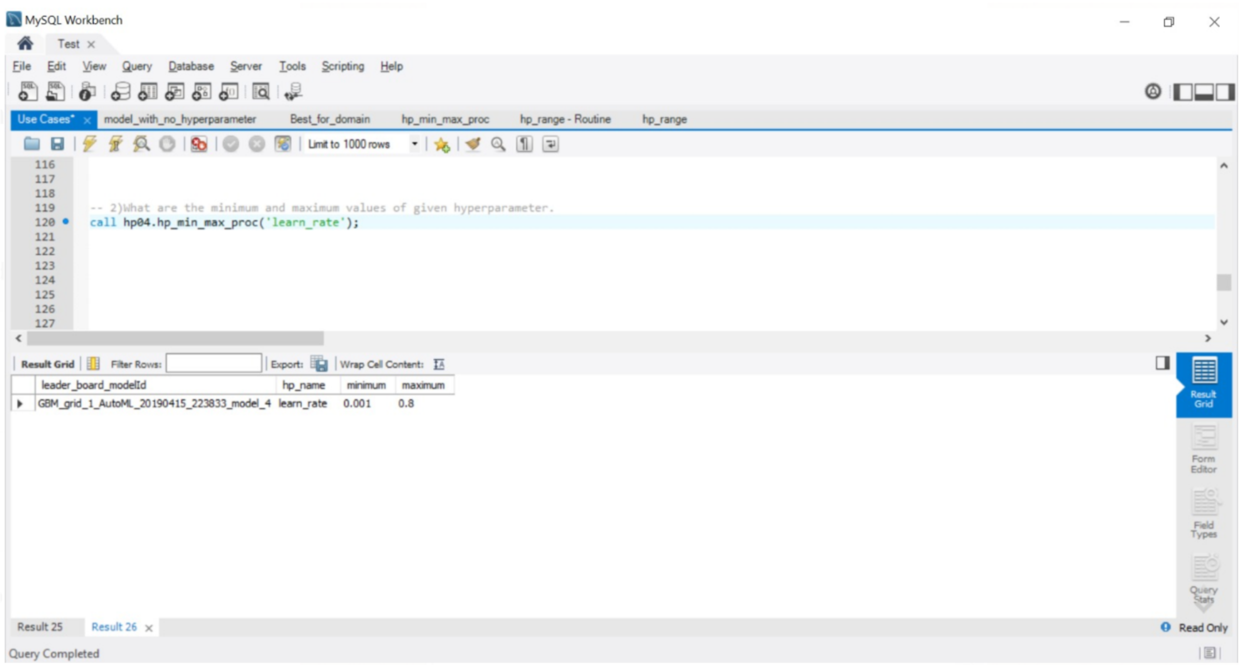
Input: Hyperparameter—learn_rate or ntrees

Output: Minimum and Maximum values of learn_rate or ntrees

CODE:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`hp_min_max_proc`(IN parameter varchar(100))
BEGIN
DECLARE default_value varchar(45);
select distinct HP04 from hp04.default_values;
IF (parameter = 'learn_rate') THEN
SELECT leader_board_modelId,'learn_rate' as hp_name,
min(learn_rate) as minimum,max(learn_rate) as maximum from
hp04.model_gbm;
ELSEIF(parameter = 'ntrees') THEN
SELECT leader_board_modelId,'ntrees' as hp_name, min(ntrees) as
minimum,max(ntrees) as maximum from hp04.model_gbm;
END IF;
```

```
END$$
DELIMITER ;
```



3. Count of all models under given algorithm and runtime.

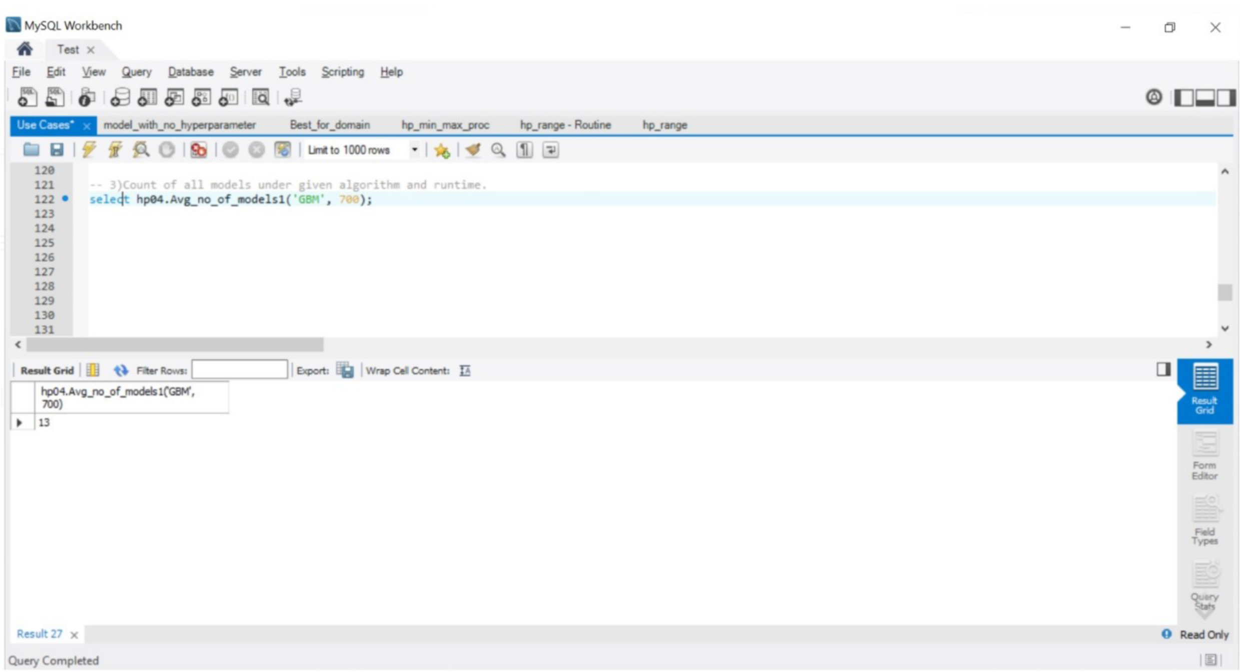
Input: Algorithm name, runtime

Output: Number of models created for that model and runtime

CODE:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` FUNCTION
`Avg_no_of_models1`(algorithm_name VARCHAR(45), run_times
INT(11)) RETURNS decimal(10,0)
DETERMINISTIC
BEGIN
DECLARE lvl decimal(10);
SELECT COUNT(*) AS Total_models INTO lvl
FROM leader_board l
JOIN run_time r ON l.run_time_runId = r.runId
WHERE LEFT(modelId,3) LIKE concat(algorithm_name,'%')
```

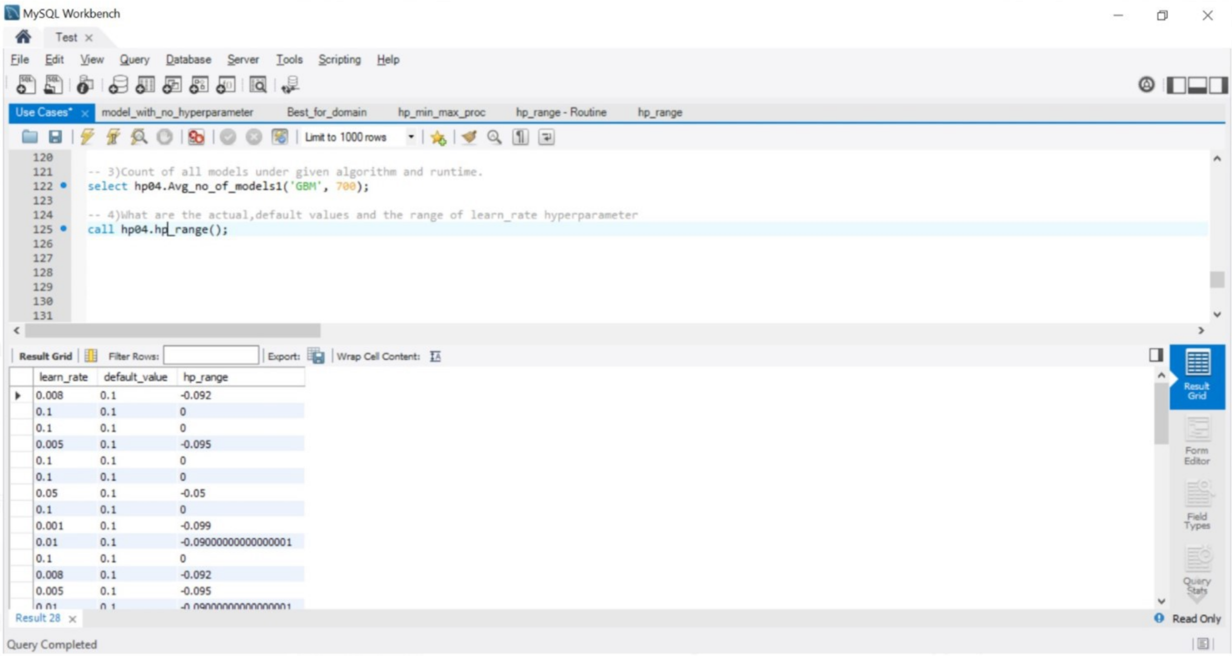
AND r.run_time = run_times;
RETURN (lvl);
END$$
DELIMITER ;



4. What are the actual, default values and the range of learn_rate hyperparameter

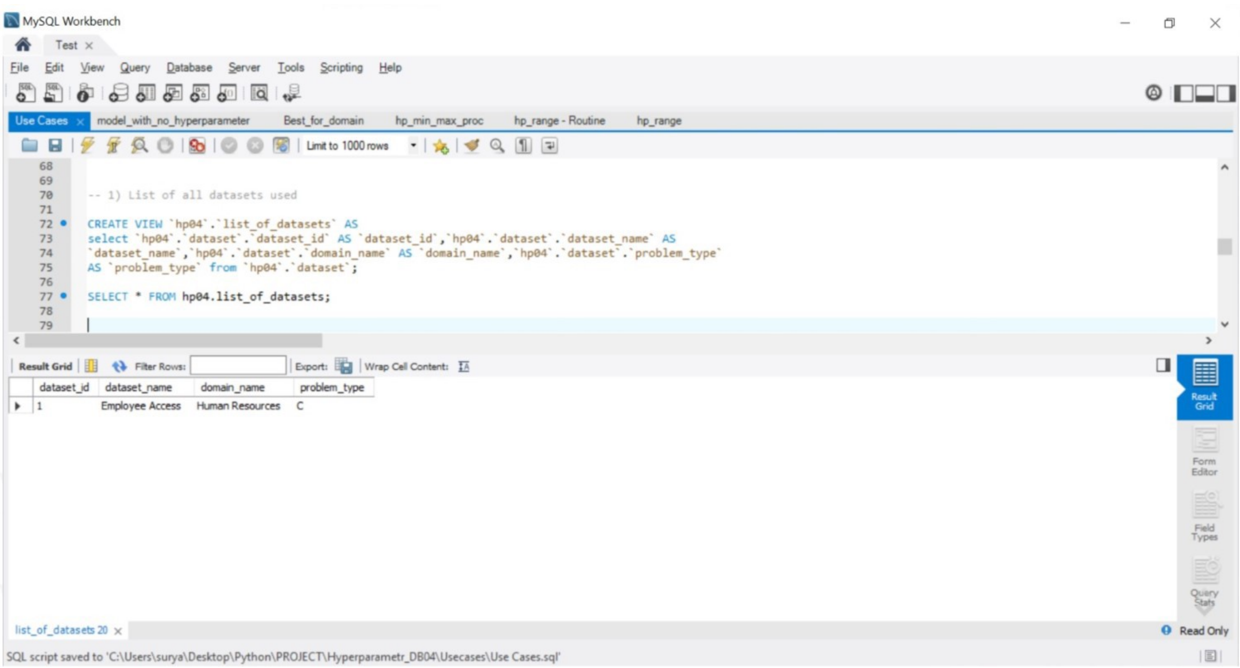
Output: All hyperparameter default values

CODE:
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `hp_range`()
BEGIN
DECLARE default_value varchar(45);
SELECT hyper_def_values INTO default_value from hp04.default_values where HP04 = 'learn_rate';
select learn_rate, default_value, (learn_rate-default_value) as hp_range from hp04.model_gbm;
END$$
DELIMITER ;

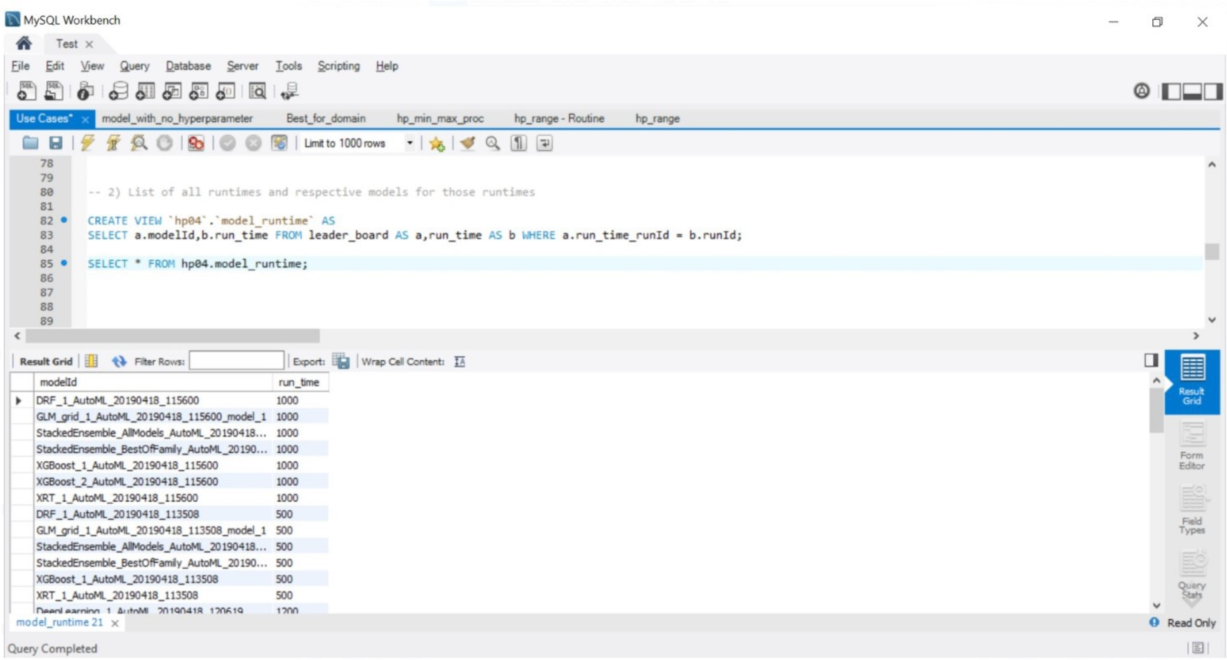


5.Views:

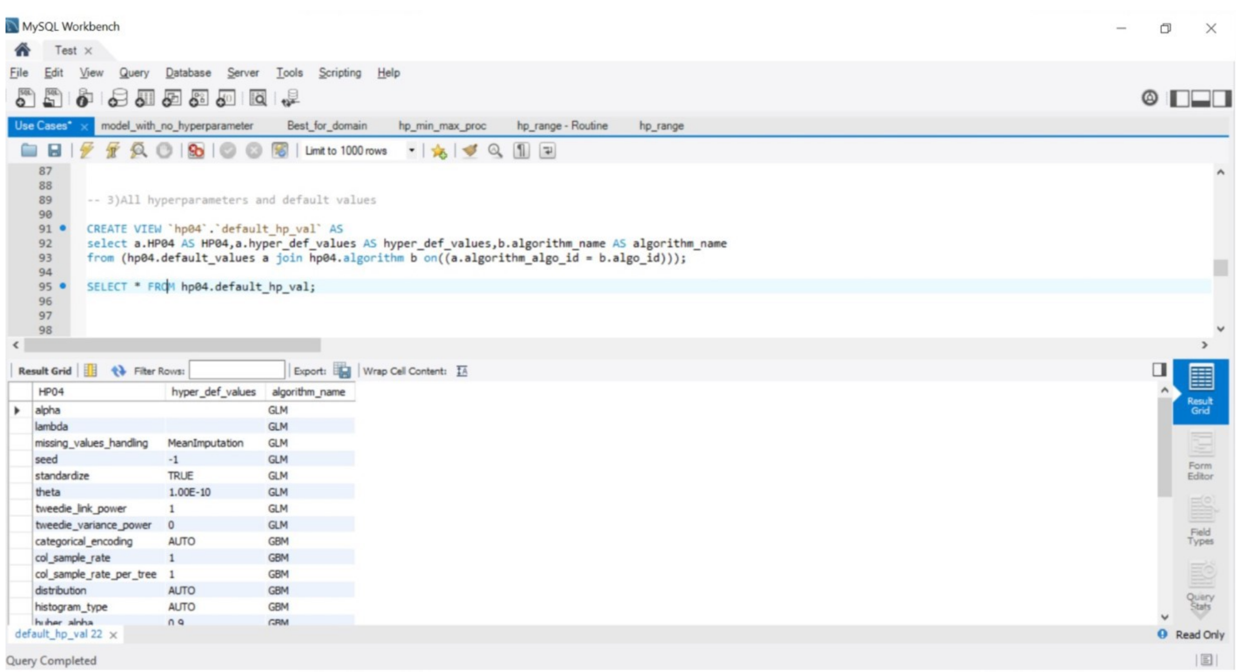
1. View of a list of all the datasets available in the database



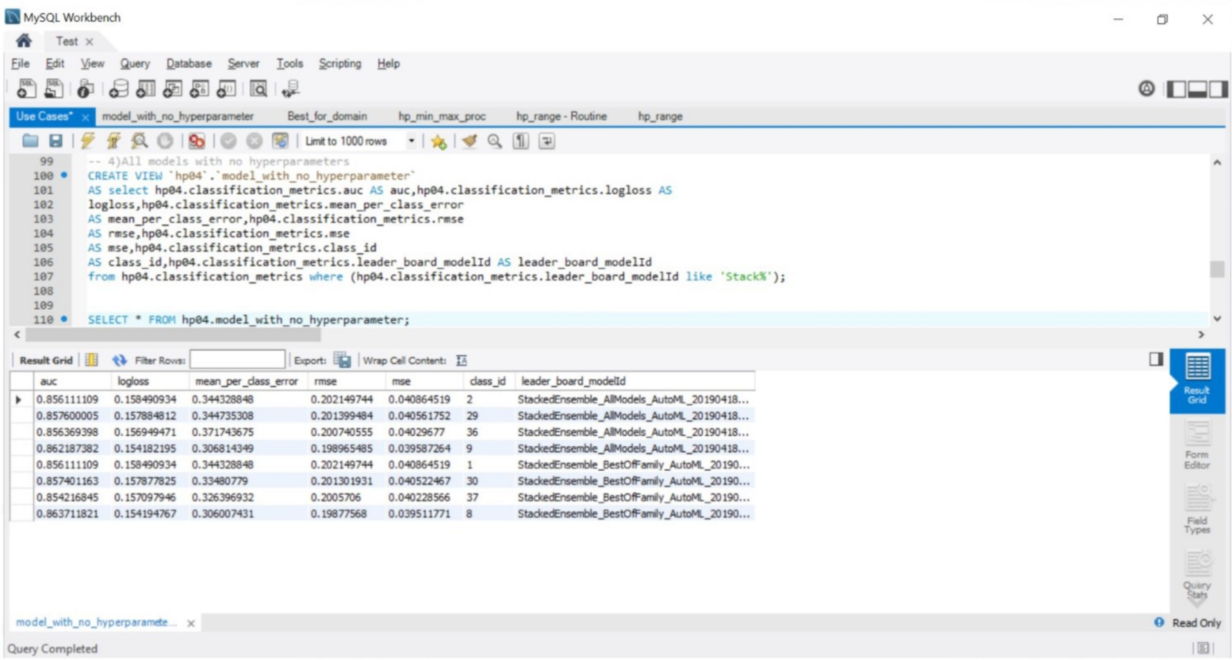
2. List of all runtimes and respective models for those runtimes



3) All hyperparameters and default values



4) All models with no hyperparameters



6. Conclusion:

Thus, after the project, we were able to create an actual physical database storing the hyperparameters' actual and default values. Through the demonstration of the use cases, we will be able to support a website for the same. The following points were covered:

1. Conceptual Diagram

2. ER—Diagram

3. Normalization

4. Creating a physical database

5. Converting JSON files into CSV files based on the model

6. Use Case preparation

7. Functions

8. Views

9. Stored Procedures

10. Documentation and Professionalism

7. Citations:

1. http://docs.h2o.ai/h2o/latest-stable/h2o-docs/grid-search.html—H2o Hyperparameters
2. https://www.hindawi.com/journals/complexity/2019/6278908/- Sampling of a Dataset in the Hyperparameter.

3. https://github.com/nikbearbrown/INFO_6210—Prof. Nik's Git Hub

4. http://www.mysqltutorial.org/mysql-stored-procedure-tutorial.aspx— Stored Procedures.

5. https://www.northeastern.edu/rise/presentations/hyperparameter-database/ - Hyperparameter Database(RISE-2019)

6. http://www.mysqltutorial.org/mysql-functions.aspx—Functions

8. License: