Hyperparameter Database Project

Team DB-04 (Tejas Munot, Manasa Vanga)

Abstract:

When an employee at any company starts work, they first need to obtain the computer access necessary to fulfill their role. It is often the case that employees figure out the access they need as they encounter roadblocks during their daily work. A supervisor then must take time out of his busy schedule to manually grant the needed access. As employees move throughout a company, this access discovery/recovery cycle wastes a nontrivial amount of time and money, which we as a team are trying to reduce.

There is a considerable amount of data regarding an employee's role within an organization and the resources to which they have access. Given the data related to current employees and their provisioned access, models can be built that automatically determine access privileges as employees enter and leave roles within a company. These auto-access models seek to minimize the human involvement required to grant or revoke employee access.

Thus, we have analyzed the data and put the results of the models and algorithms into a physical database. The practical use cases, functions and stored procedures

Objective:

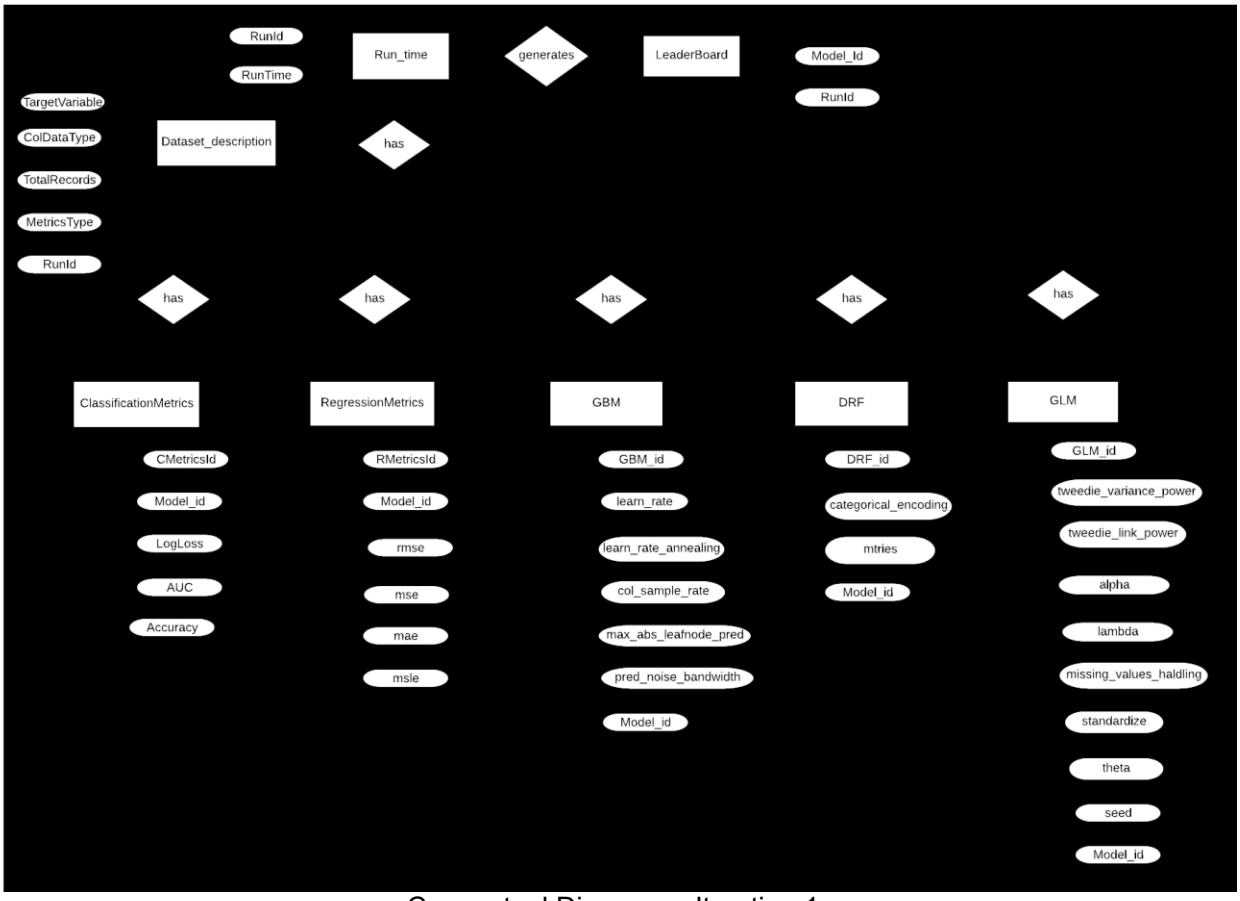
The objective of this project is to build a physical database storing details and values of the various hyperparameters generated through running the dataset into H2O. Finally, this database would help support a website which would help data enthusiasts get the best hyperparameter values for their respective datasets.

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.

Here's how the process flow looks for the team:

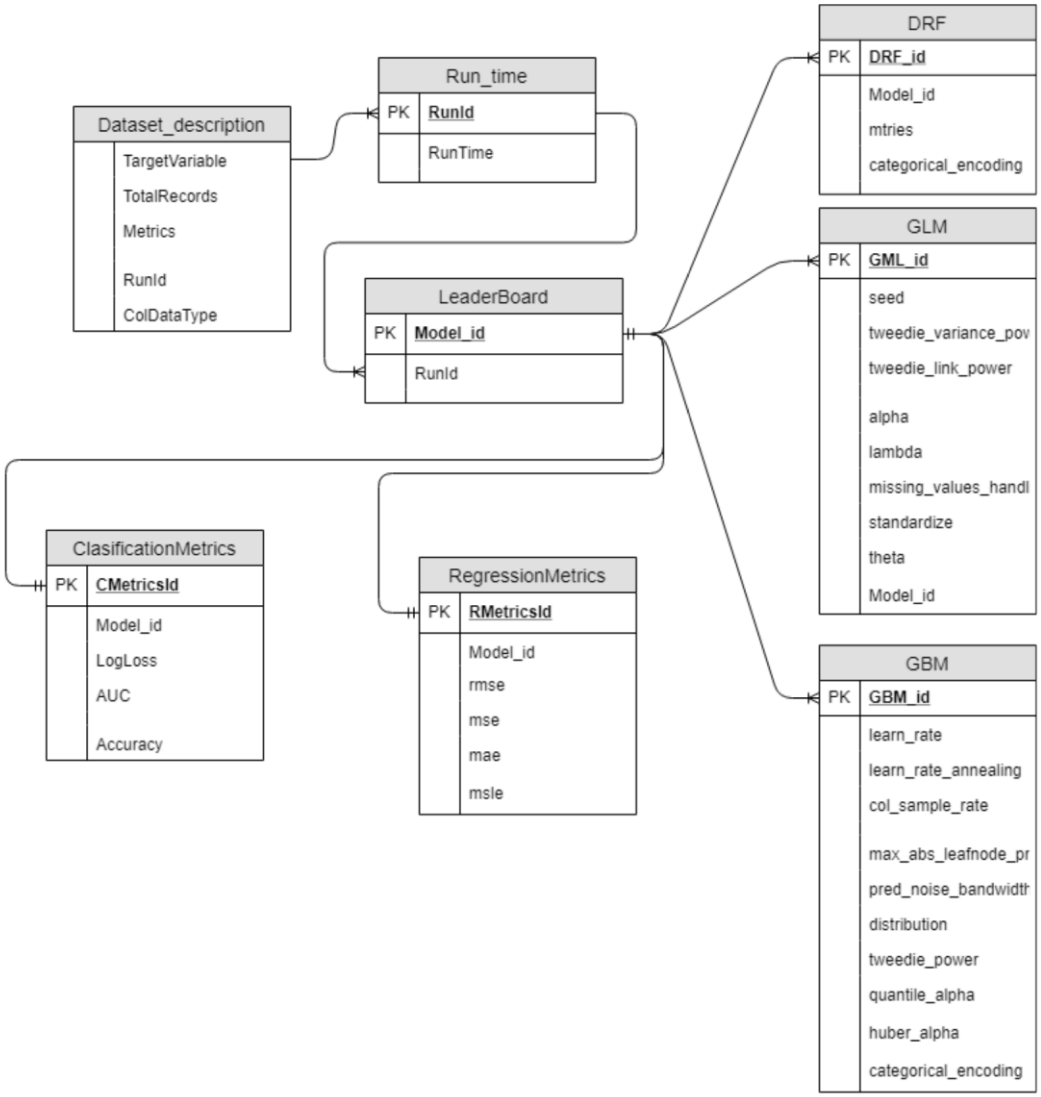
1. Conceptual Diagram
2. ER – Diagram
3. Normalization
4. Revised ER – Diagram
5. Converting JSON files into csv files based on the model
6. Creating the physical database
7. Writing practical use cases
8. Documentation and Professionalism

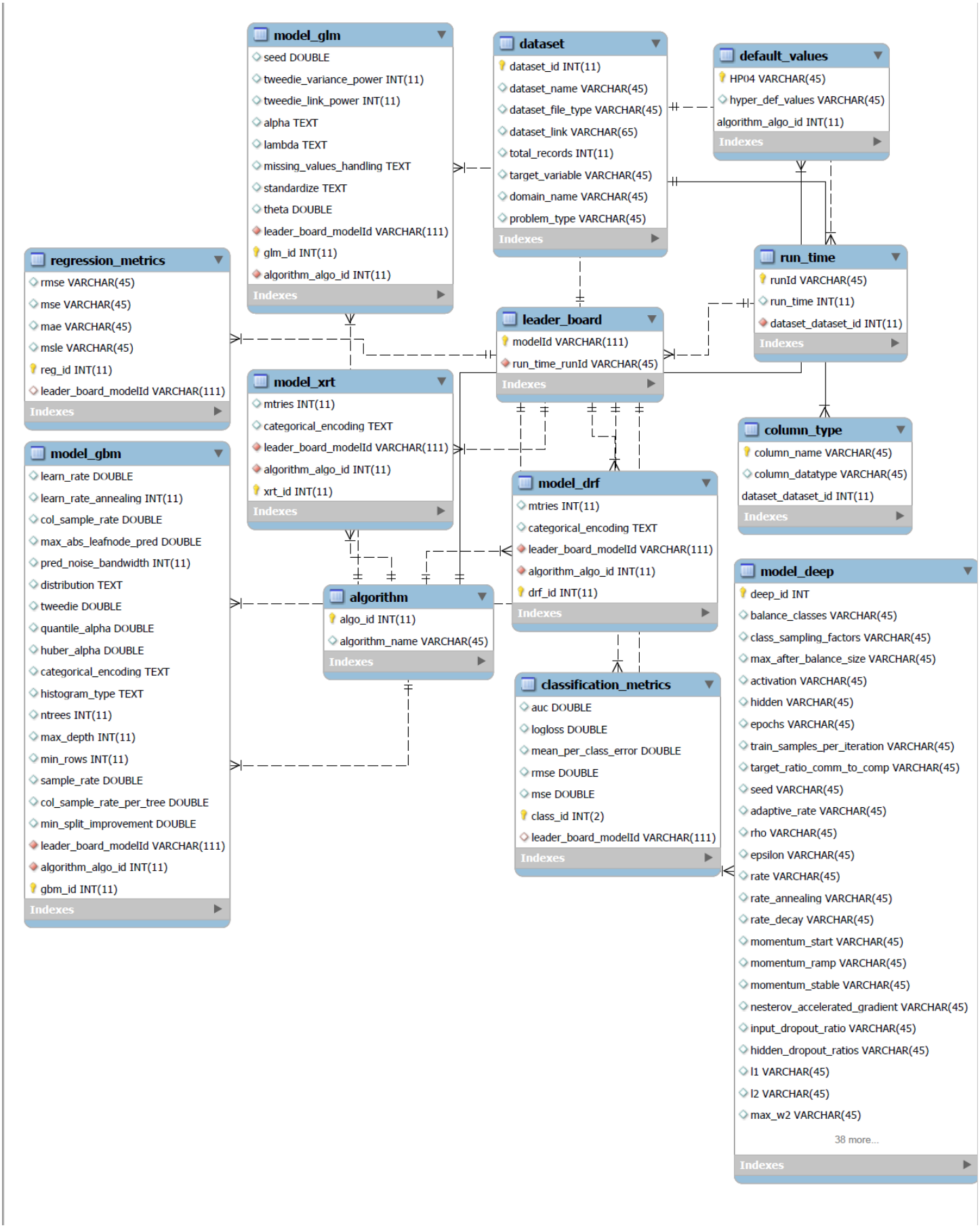
The objective for the loading the data in the physical database was to convert the JSON files into comma-separated value files (csv's). We did that using python. We then identified the entities, attributes, and relationships. Thus, we came up with the first version of our conceptual diagram.



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. After a lot of iterations, we came up with a final Entity Relationship Diagram as follows:


Dataset_description
TargetVariable
TotalRecords
Metrics
RunId
ColDataType
Run_time
PK
RunId
RunTime
LeaderBoard
PK
Model_id
RunId
DRF
PK
DRF_id
Model_id
mtries
categorical_encoding
GLM
PK
GML_id
seed
tweedie_variance_pov
tweedie_link_power
alpha
lambda
missing_values_handl
standardize
theta
Model_id
GBM
PK
GBM_id
learn_rate
learn_rate_annealing
col_sample_rate
max_abs_leafnode_pr
pred_noise_bandwidth
distribution
tweedie_power
quantile_alpha
huber_alpha
categorical_encoding
ClasificationMetrics
PK
CMetricsId
Model_id
LogLoss
AUC
Accuracy
RegressionMetrics
PK
RMetricsId
Model_id
rmse
mse
mae
msle



ERD final image

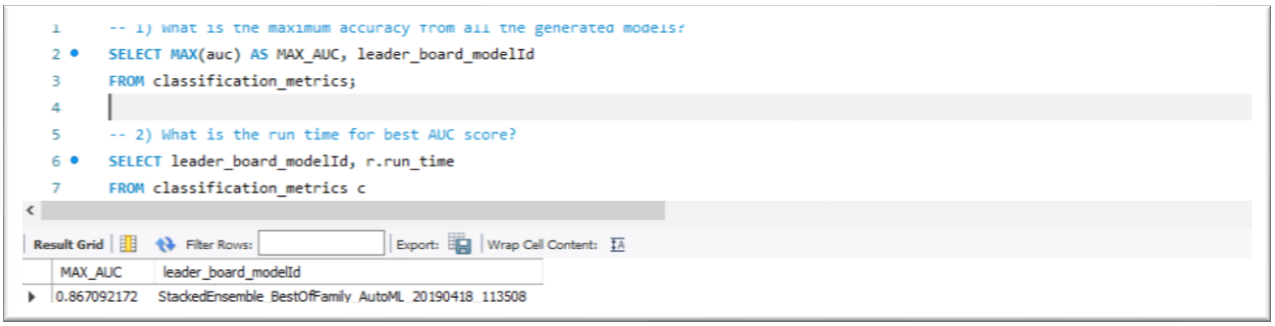
We were now ready to create the physical database. We used MySQL Workbench extensively. All our queries are written and executed in MySQL Workbench.

Use Cases:

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 best AUC score?

SELECT leader_board_modelId, r.run_time

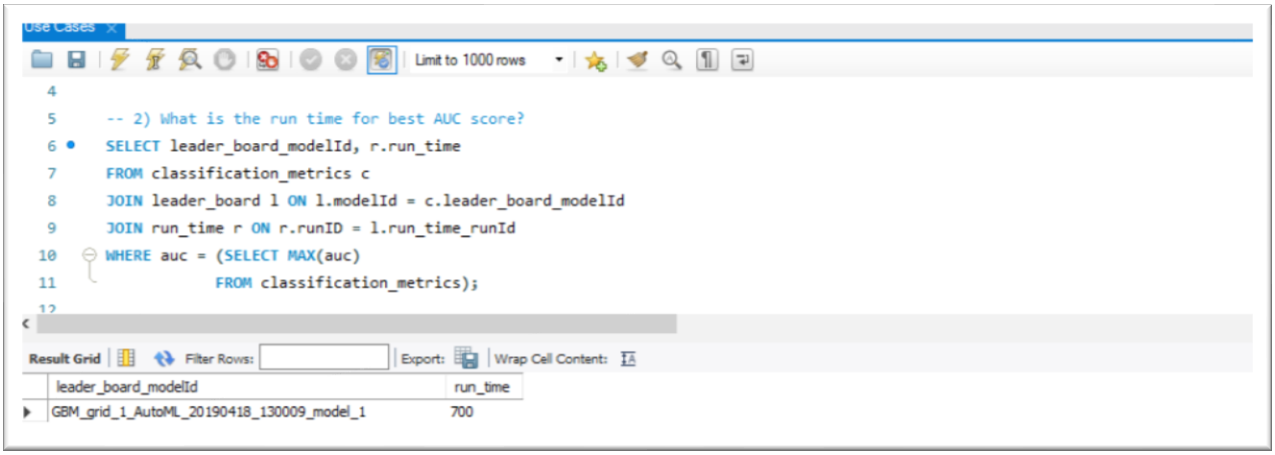
FROM classification_metrics c

JOIN leader_board l ON l.modelId = c.leader_board_modelId

JOIN run_time r ON r.runID = l.run_time_runId

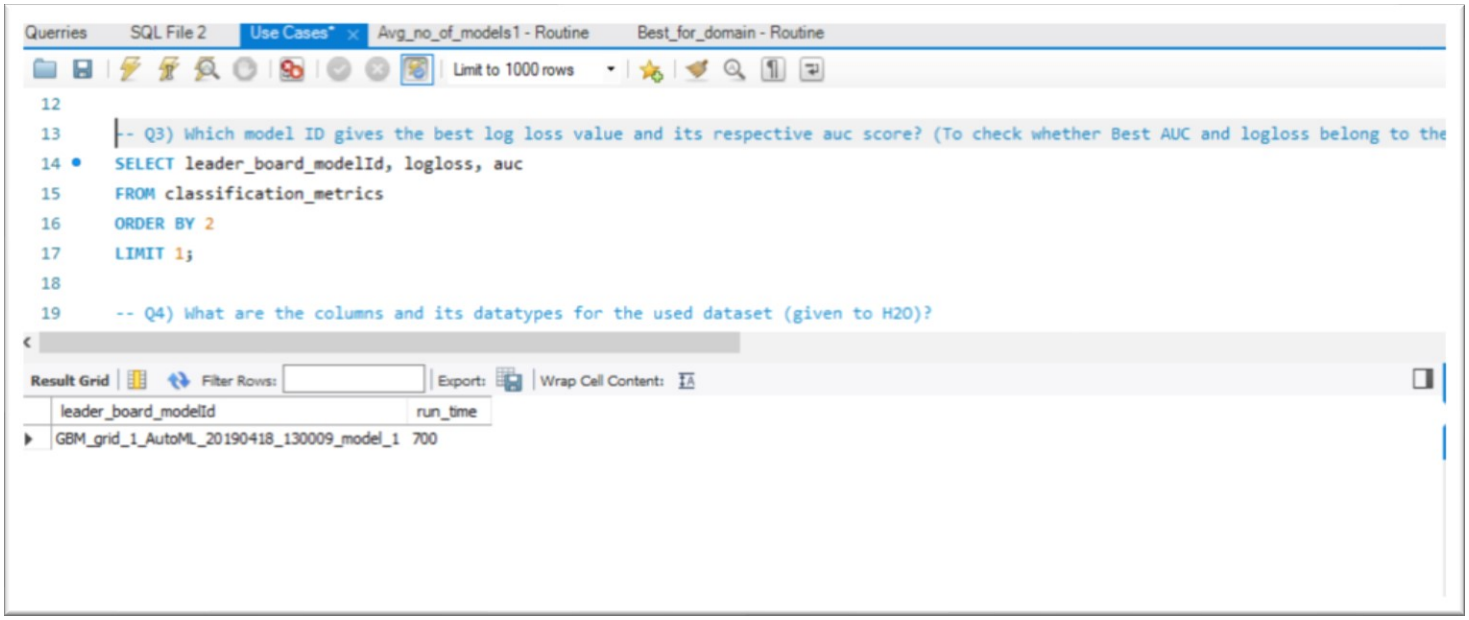
WHERE auc = (SELECT MAX(auc)

FROM classification_metrics);



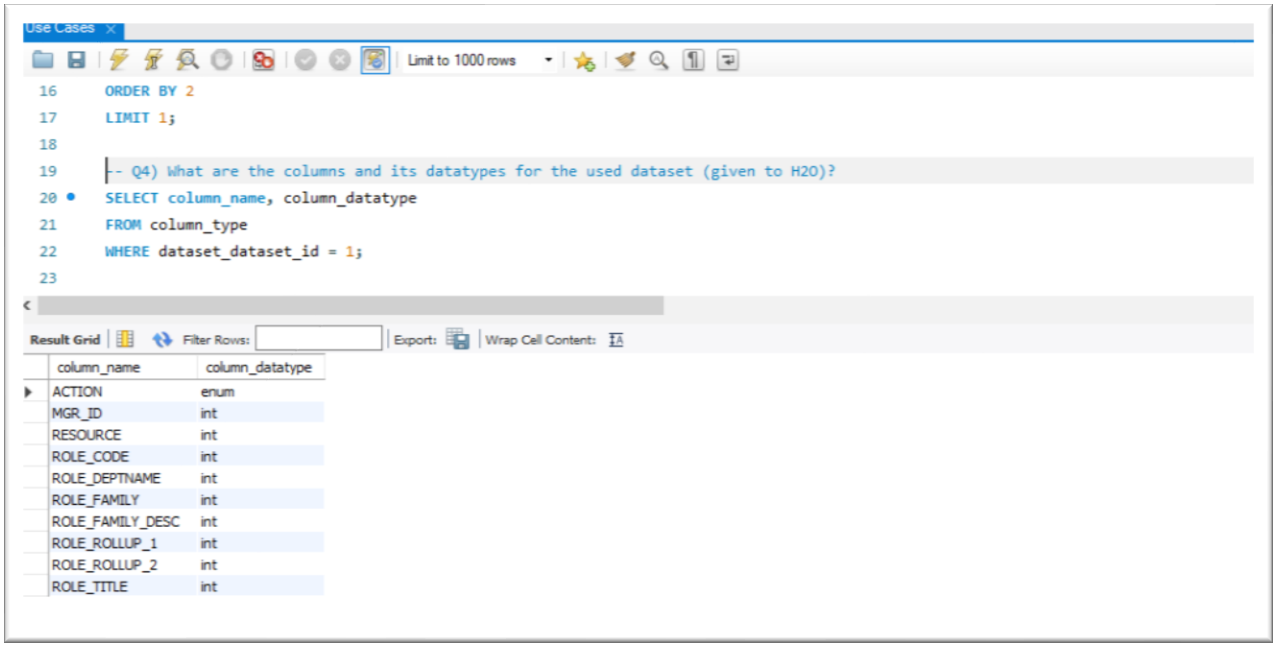
Q3) Which model ID gives the best log loss value? (To check whether Best AUC and logloss belong to the same model?)

```
SELECT leader_board_modelId, logloss, auc

FROM classification_metrics

ORDER BY 2

LIMIT 1;
```



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;
```



Q5) What is the best algorithm for classification type?

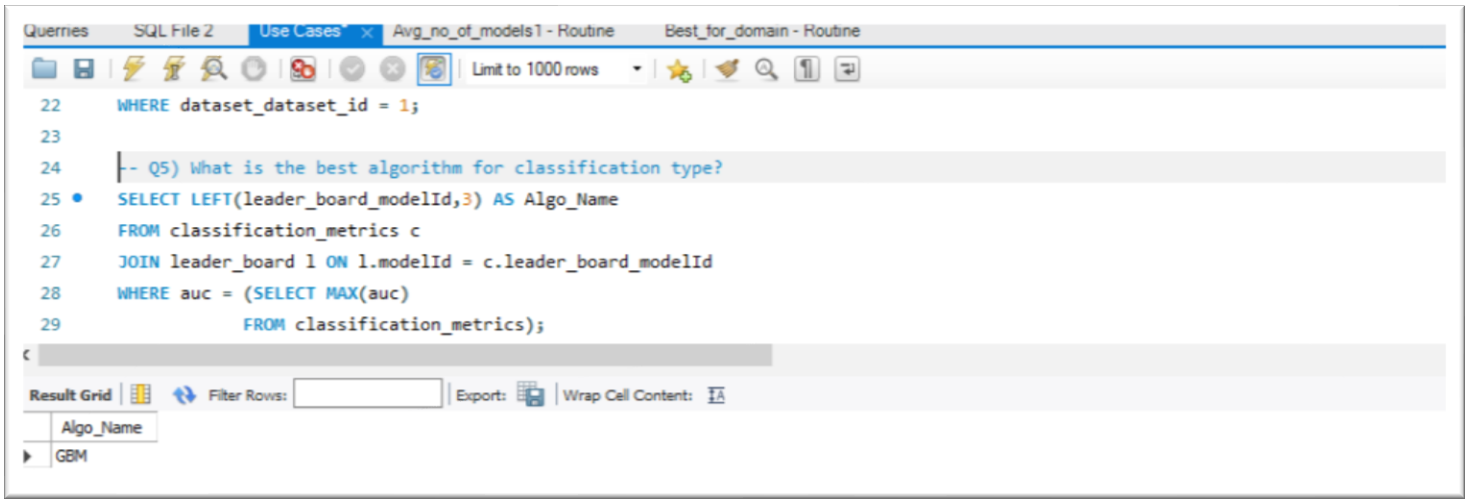
SELECT LEFT(leader_board_modelId,3) AS Algo_Name

FROM classification_metrics c

JOIN leader_board l ON l.modelId = c.leader_board_modelId

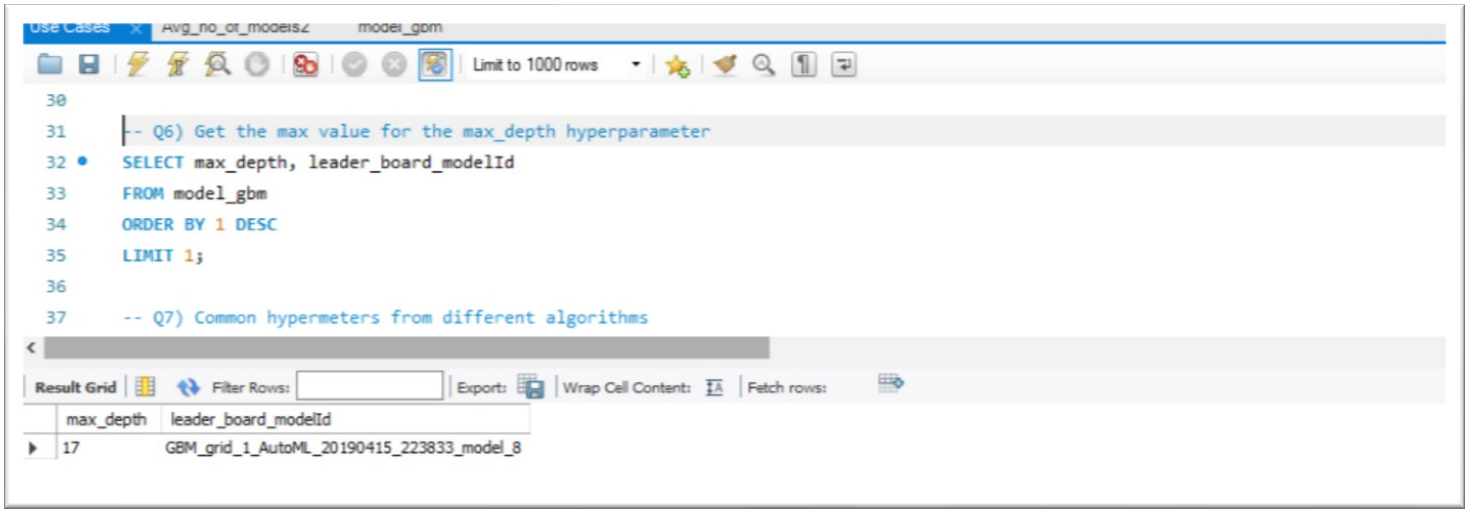
WHERE auc = (SELECT MAX(auc)

FROM classification_metrics);



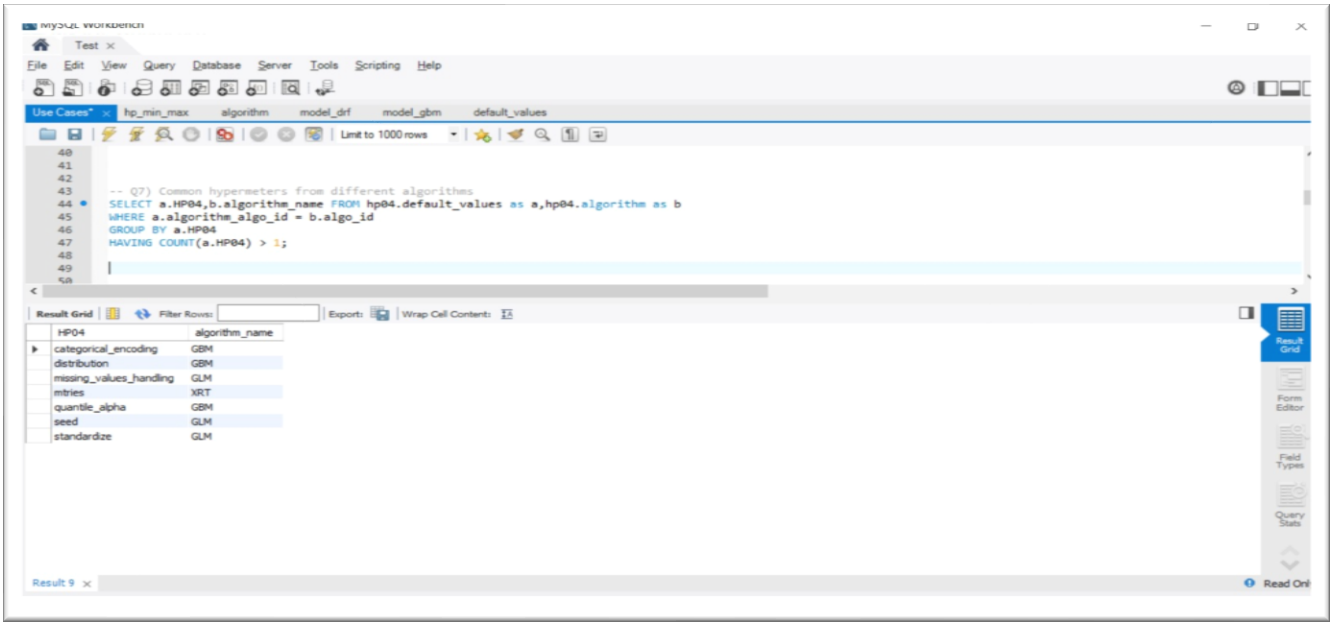
Q6) Get the Max value of the max_depth hyperparameter

```
SELECT max_depth, leader_board_modelId

FROM model_gbm

ORDER BY 1 DESC

LIMIT 1;
```



Q7) Common hypermeters from different algorithms

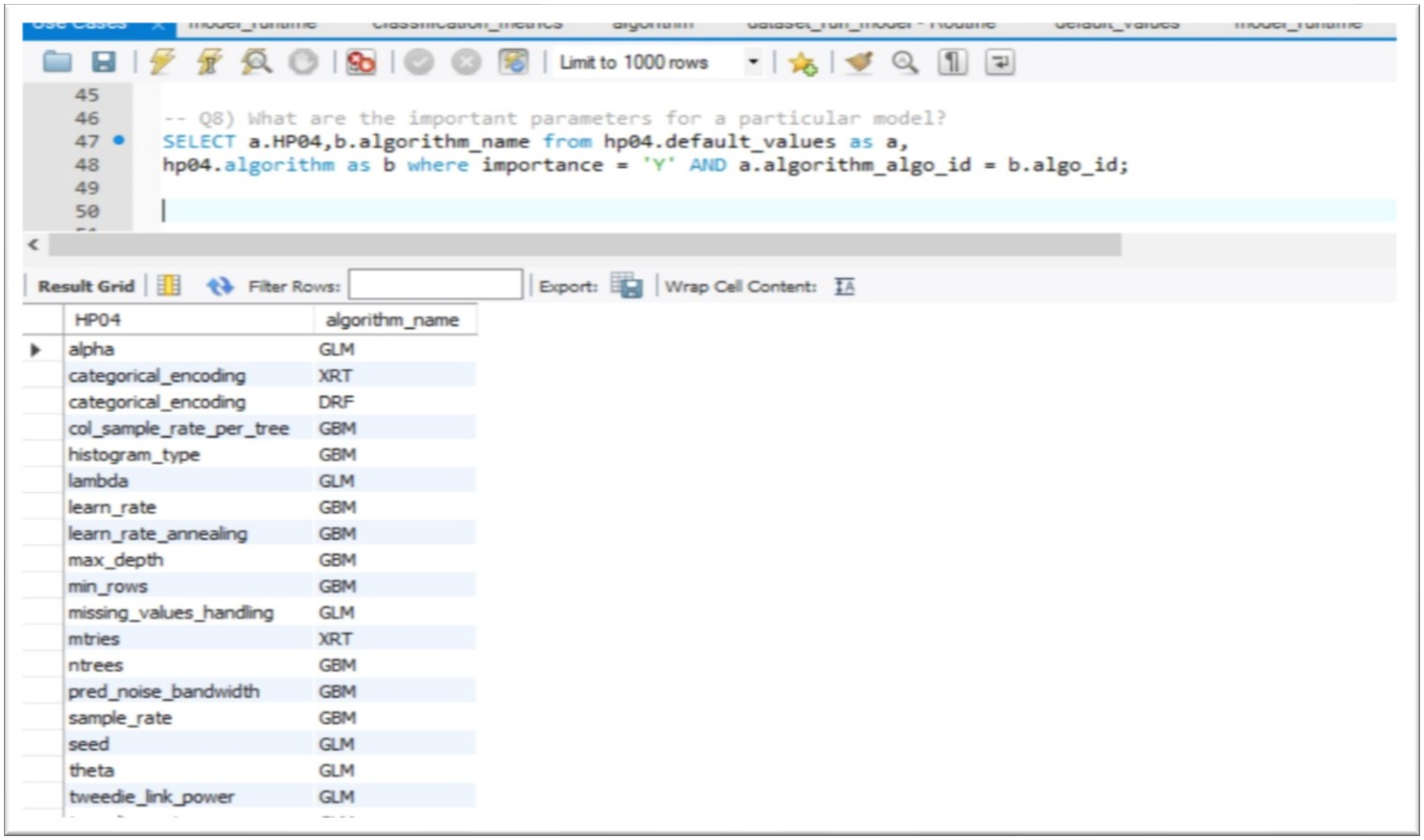
```
SELECT a.HP04,b.algorithm_name FROM hp04.default_values as a,hp04.algorithm as b

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;



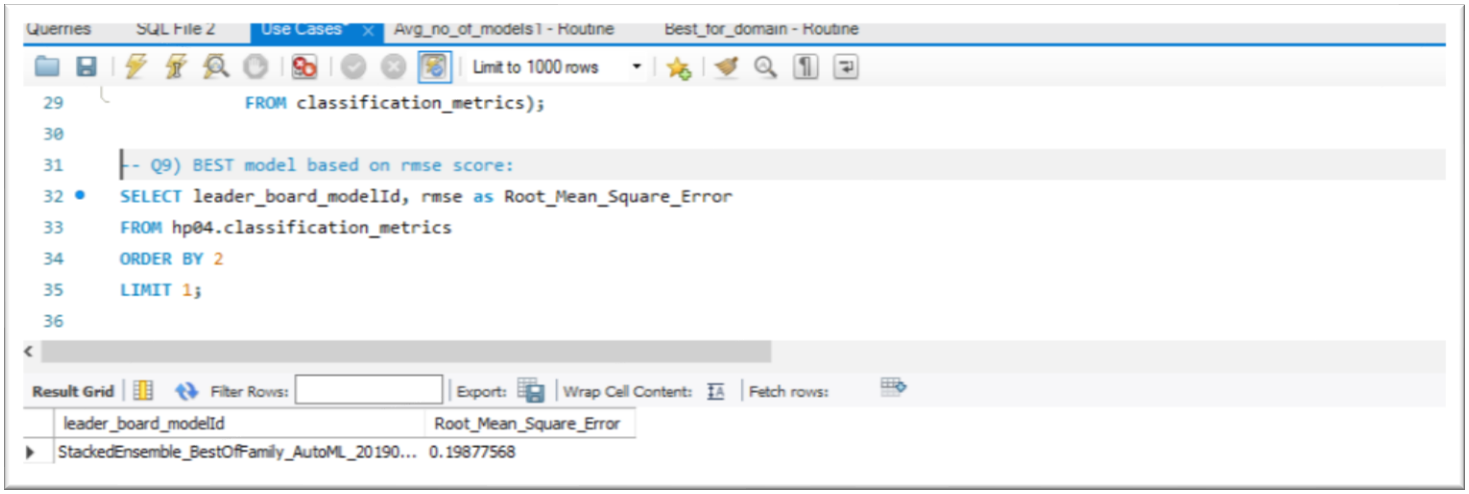
Q9) BEST model based on rmse score:

SELECT leader_board_modelId, rmse as Root_Mean_Square_Error

FROM hp04.classification_metrics

ORDER BY 2

LIMIT 1;



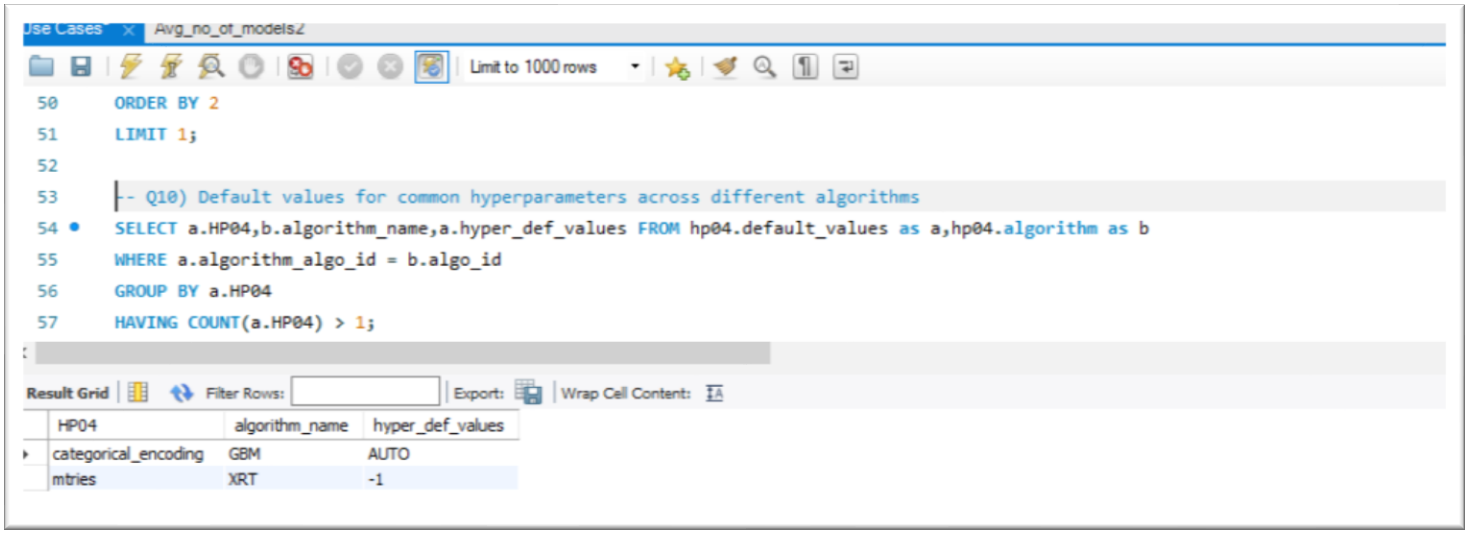
Q10) Default values for common hyperparameters across different algorithms

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

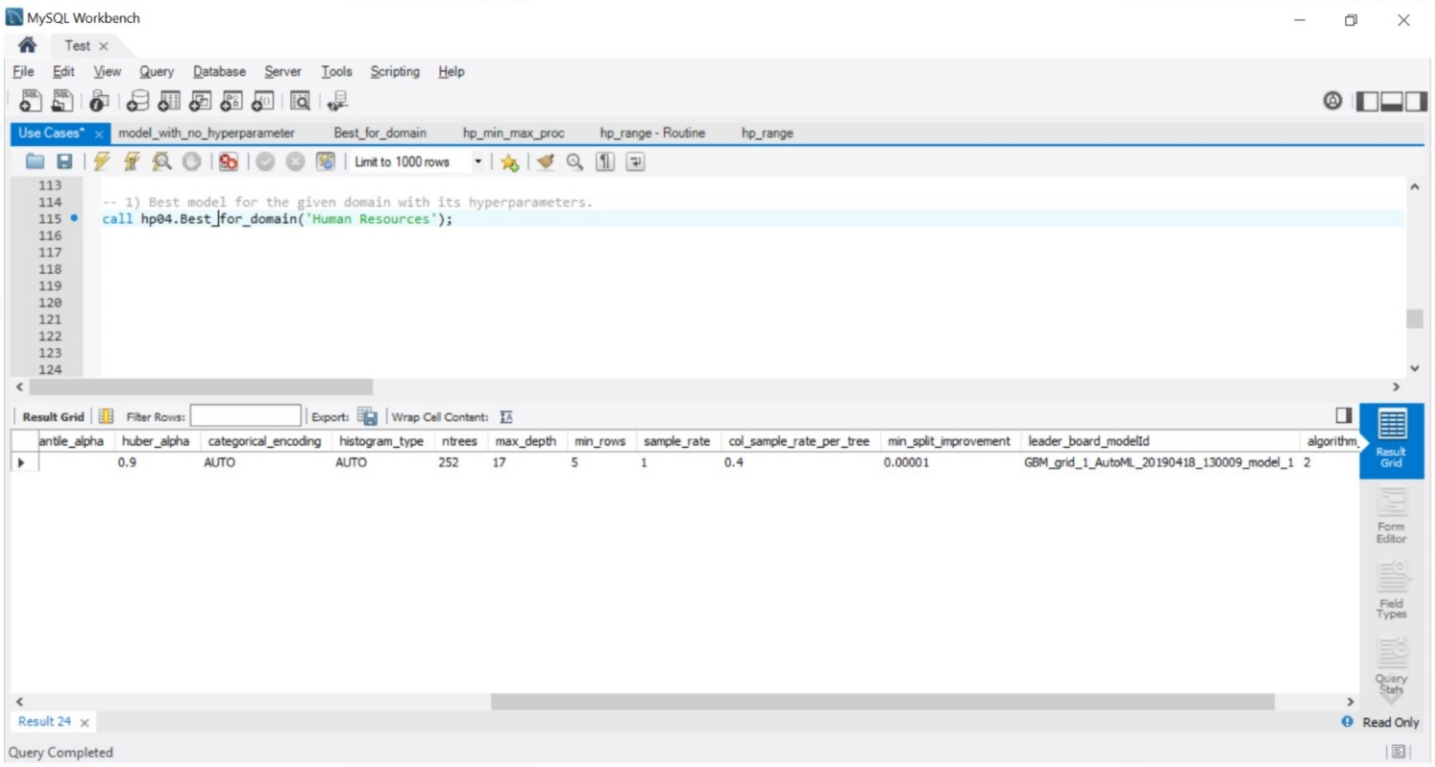
HAVING COUNT(a.HP04) > 1;



Functions and stored procedures:

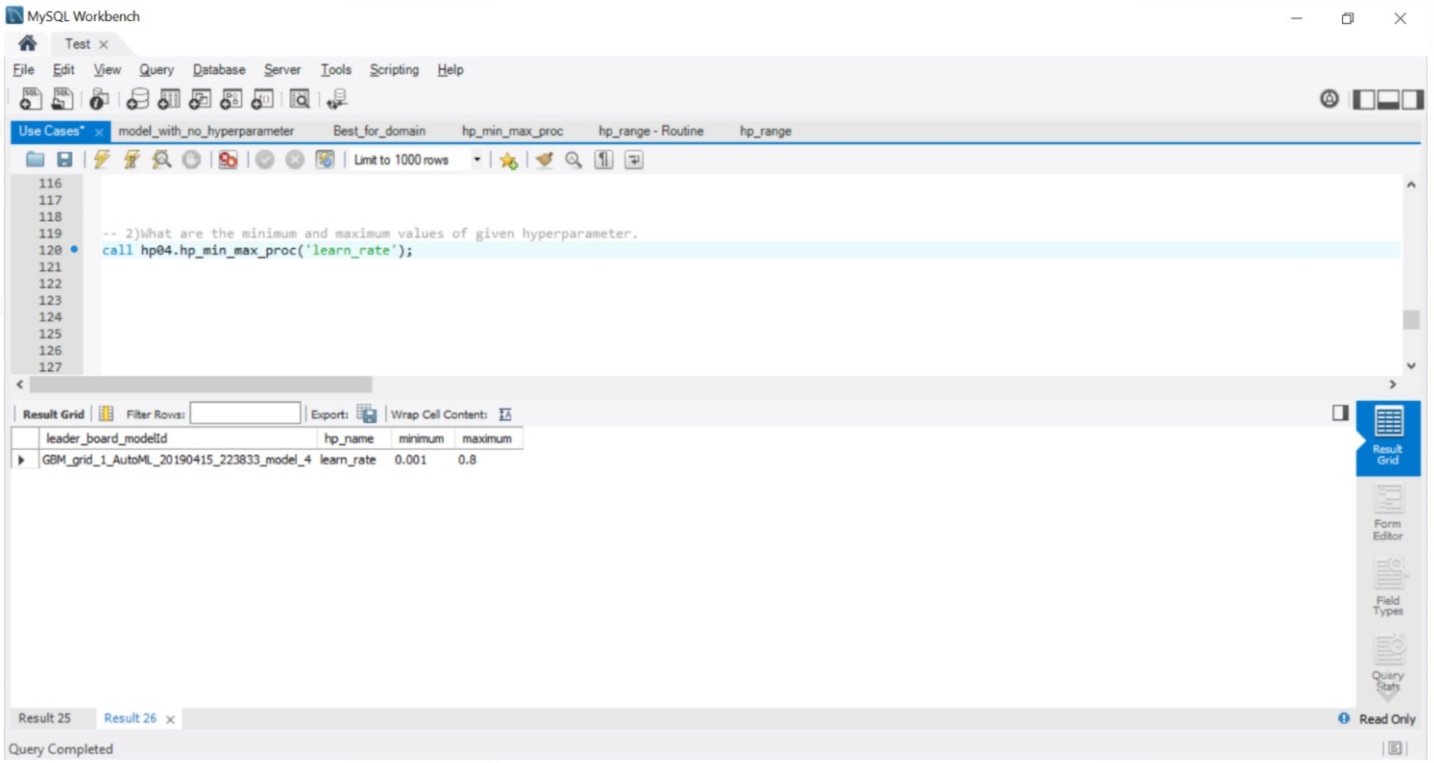
1. Best model for the given domain with its hyperparameters.
   Input: Domain Name
   Output: The best model with its actual hyperparameter values
   CODE:

```
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 ;
```



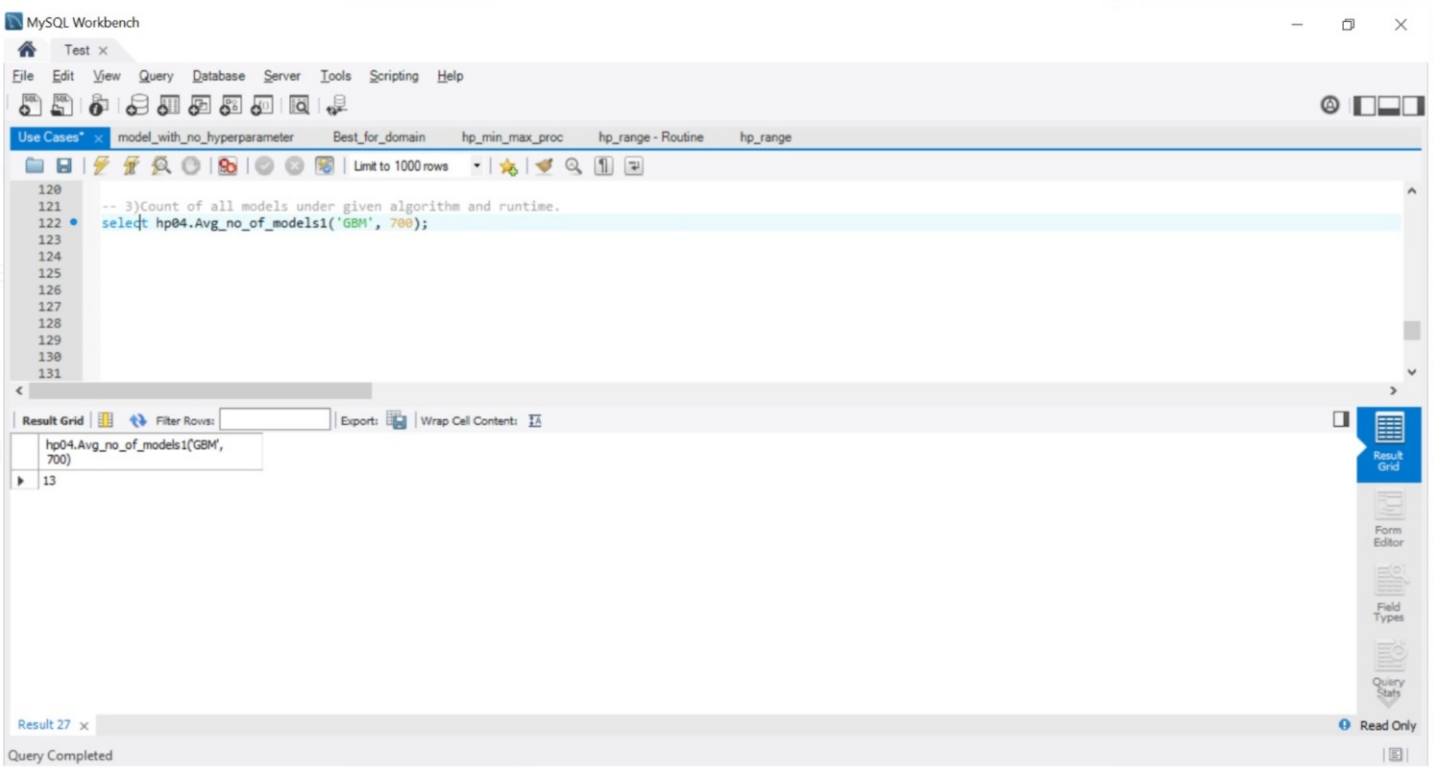
2. What are the minimum and maximum values of learn_rate, ntrees hyperparameter?
   Input: Hyperparameter – learn_rate or ntrees
   Output: Minimum and Maximum values of learn_rate or ntrees
   CODE:

```
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:

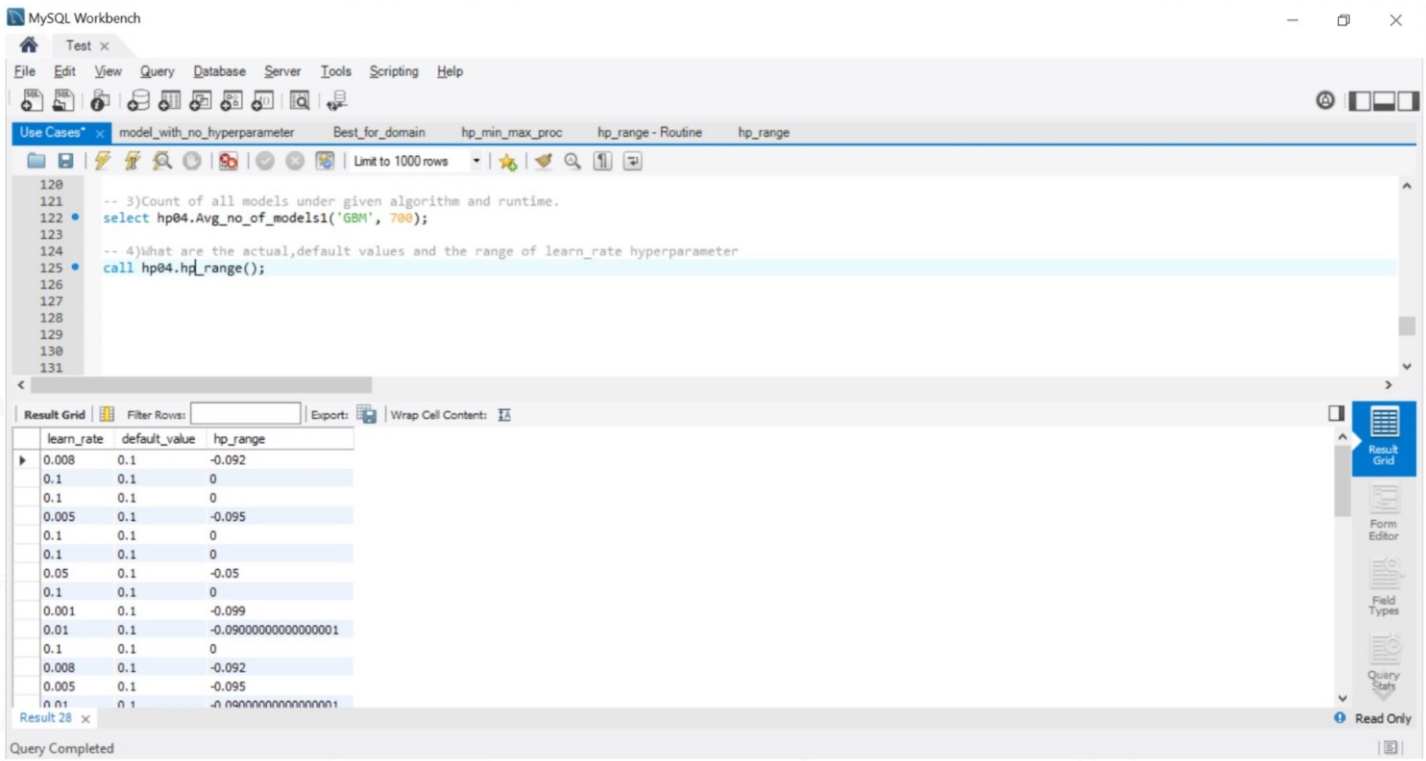
```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` FUNCTION `Avg_no_of_models1`(algorithm_name VARCHAR(45), run_times INT(11)) RETURNS decimal(10,0)
    DETERMINISTIC
BEGIN
    DECLARE lvl decimal(10);

    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

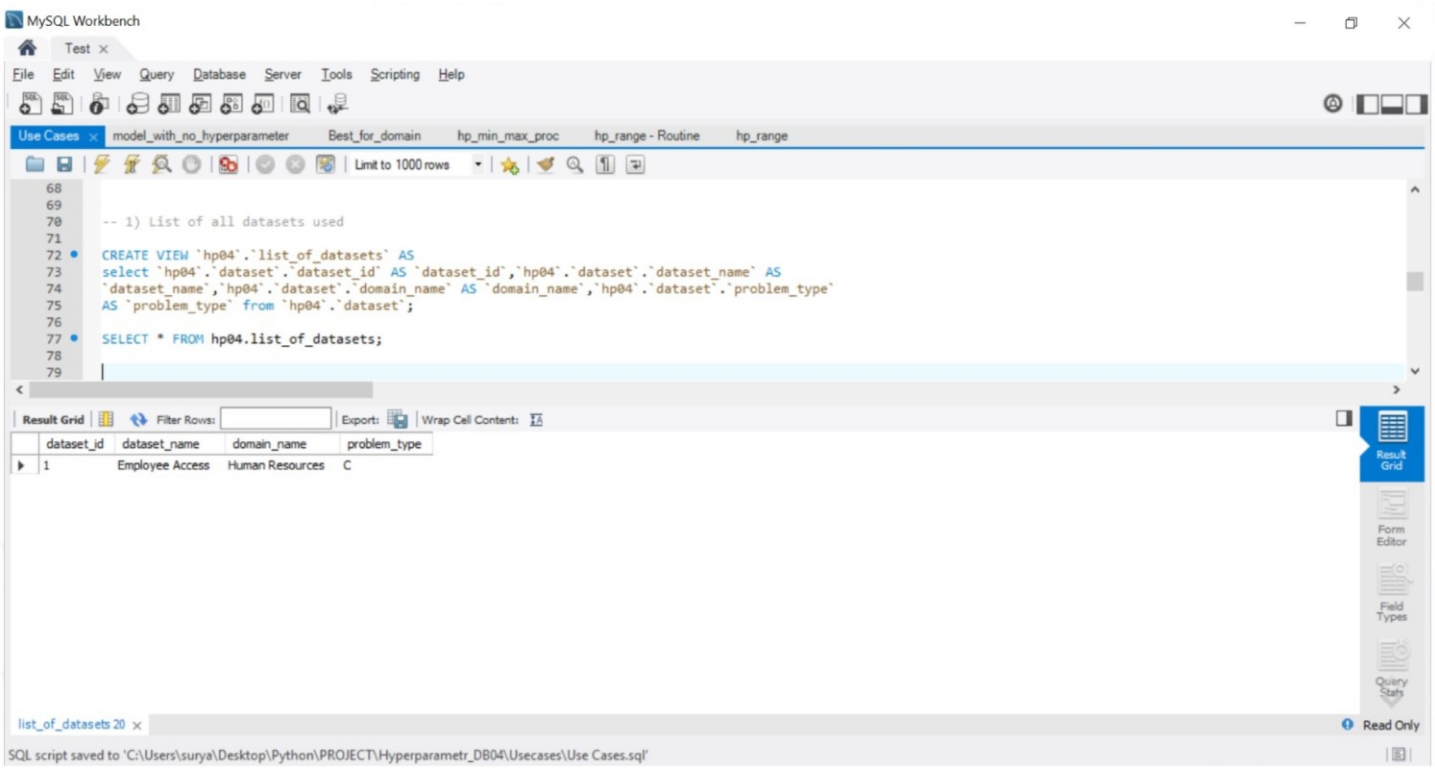
   CODE:

```
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 ;
```

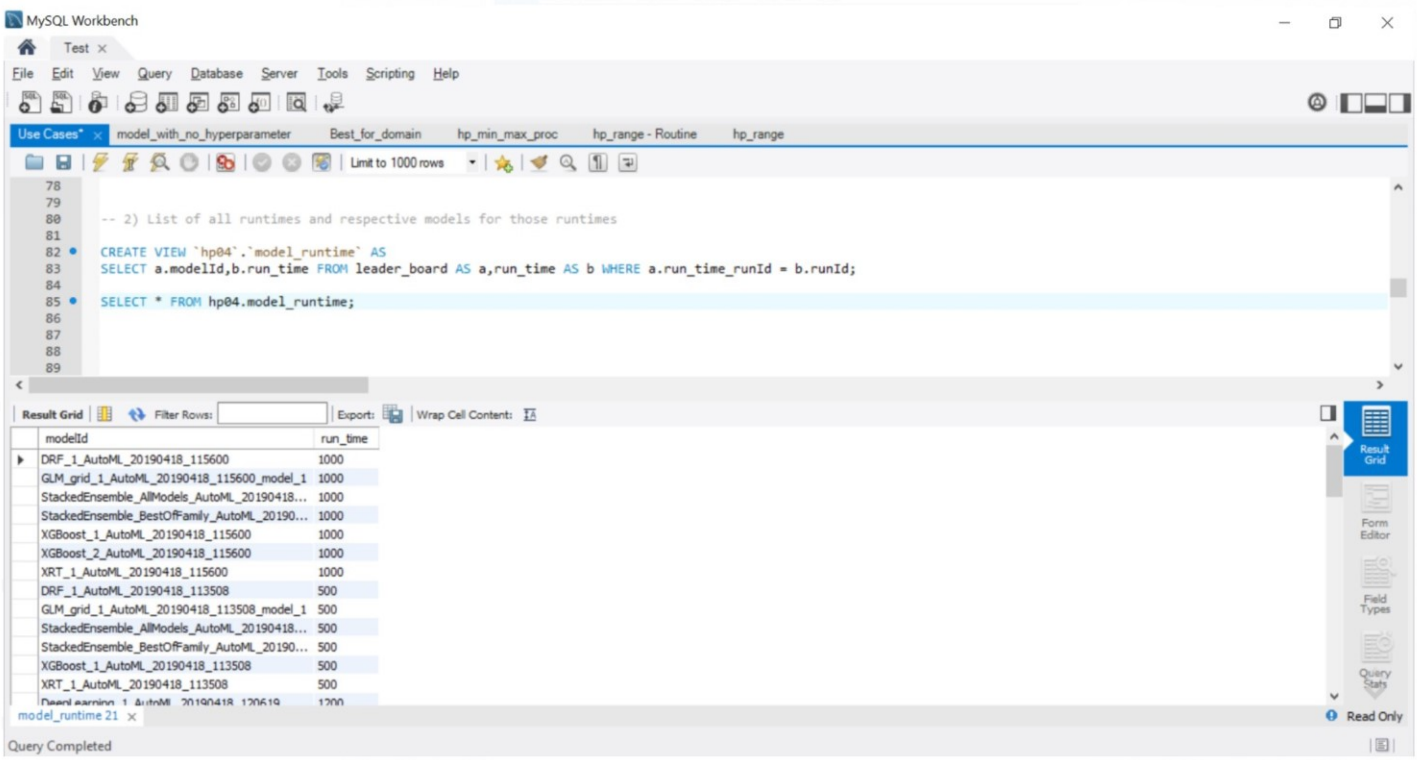


Views:

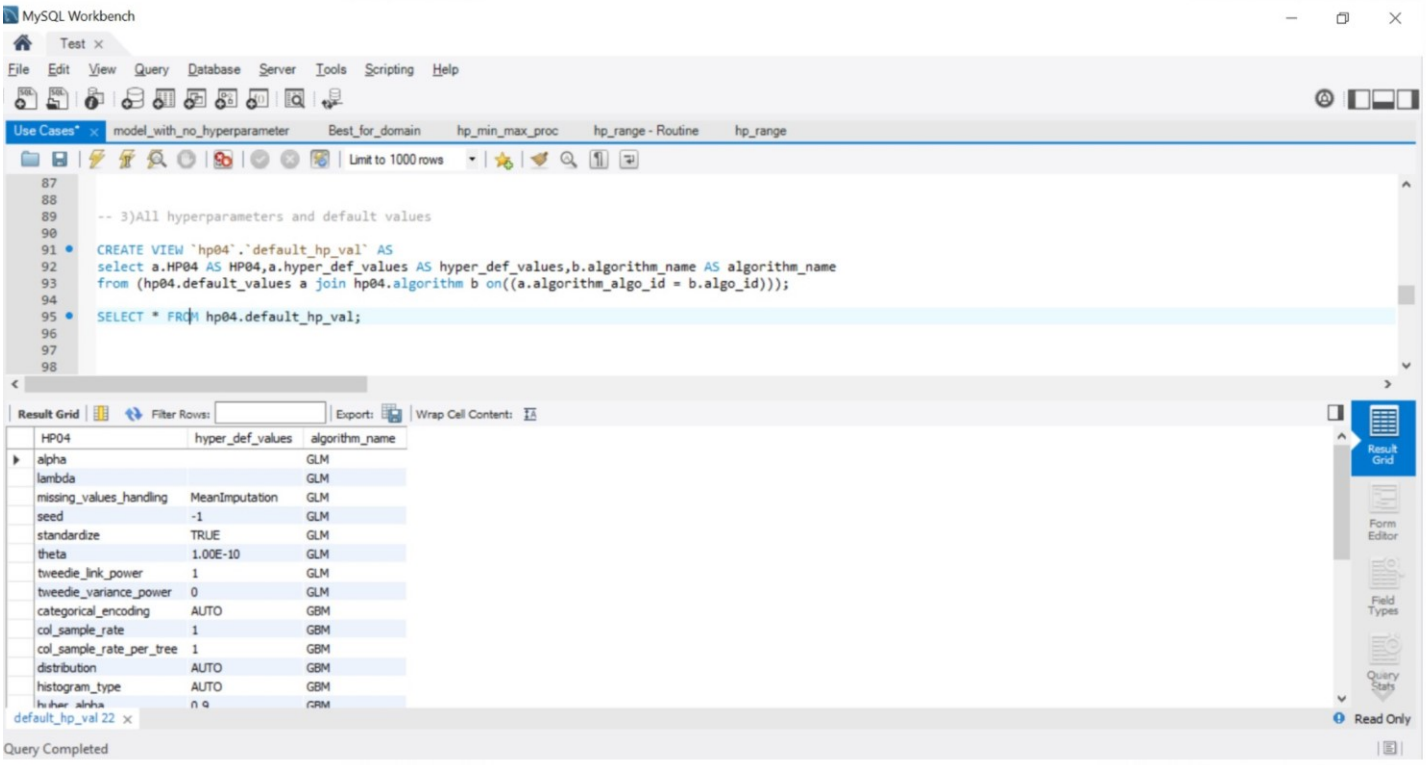
1. View of a list of all the datasets available in the database



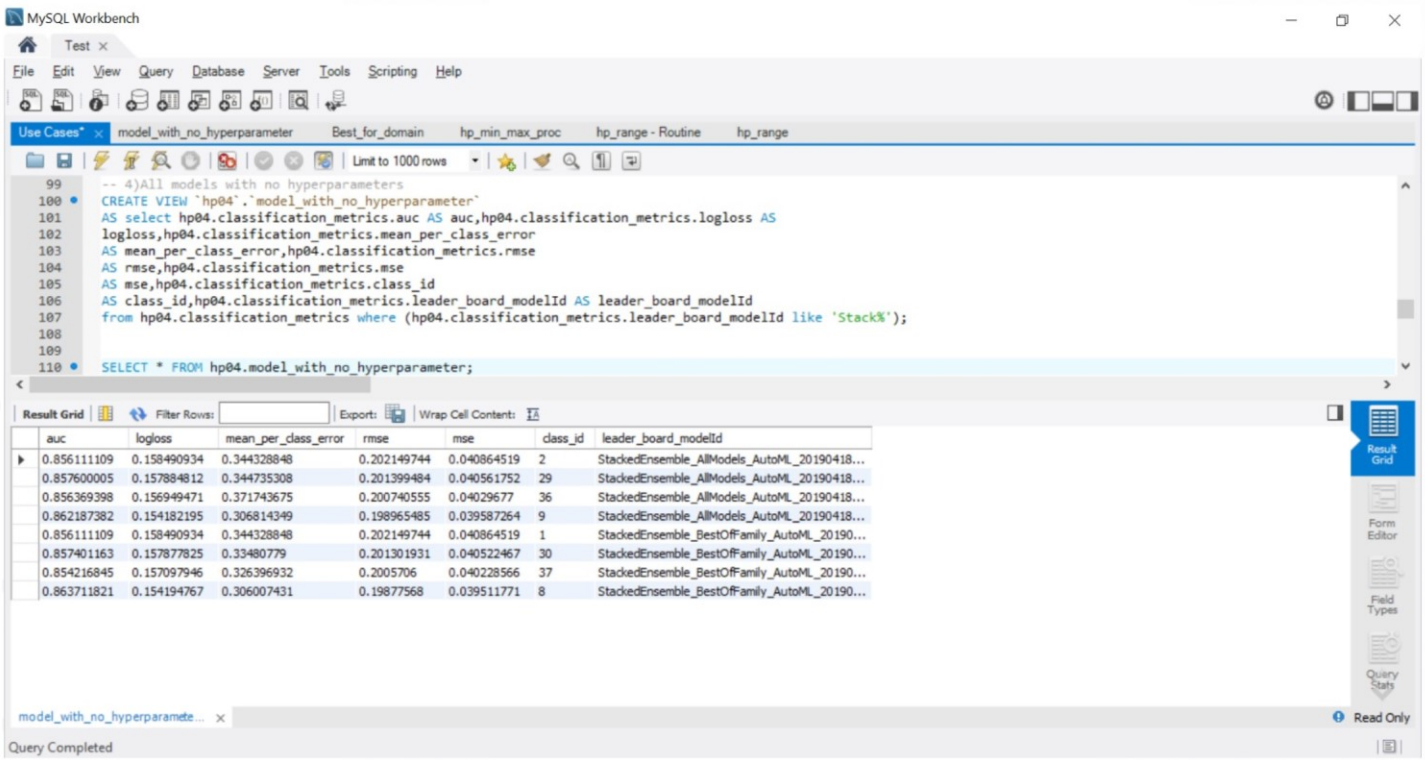
2. List of all runtimes and respective models for those runtimes



3) All hyperparameters and default values



4) All models with no hyperparameters



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. Use Case preparation
6. Functions
7. Views
8. Stored Procedures

Citations:

1. http://docs.h2o.ai/h2o/latest-stable/h2o-docs/grid-search.html - H20 Hyperparameters
2. https://github.com/nikbearbrown/INFO_6210 - Prof. Nik's Git Hub
3. http://www.mysqltutorial.org/mysql-stored-procedure-tutorial.aspx – Stored Procedures
4. http://www.mysqltutorial.org/mysql-functions.aspx – Functions

License: