

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:

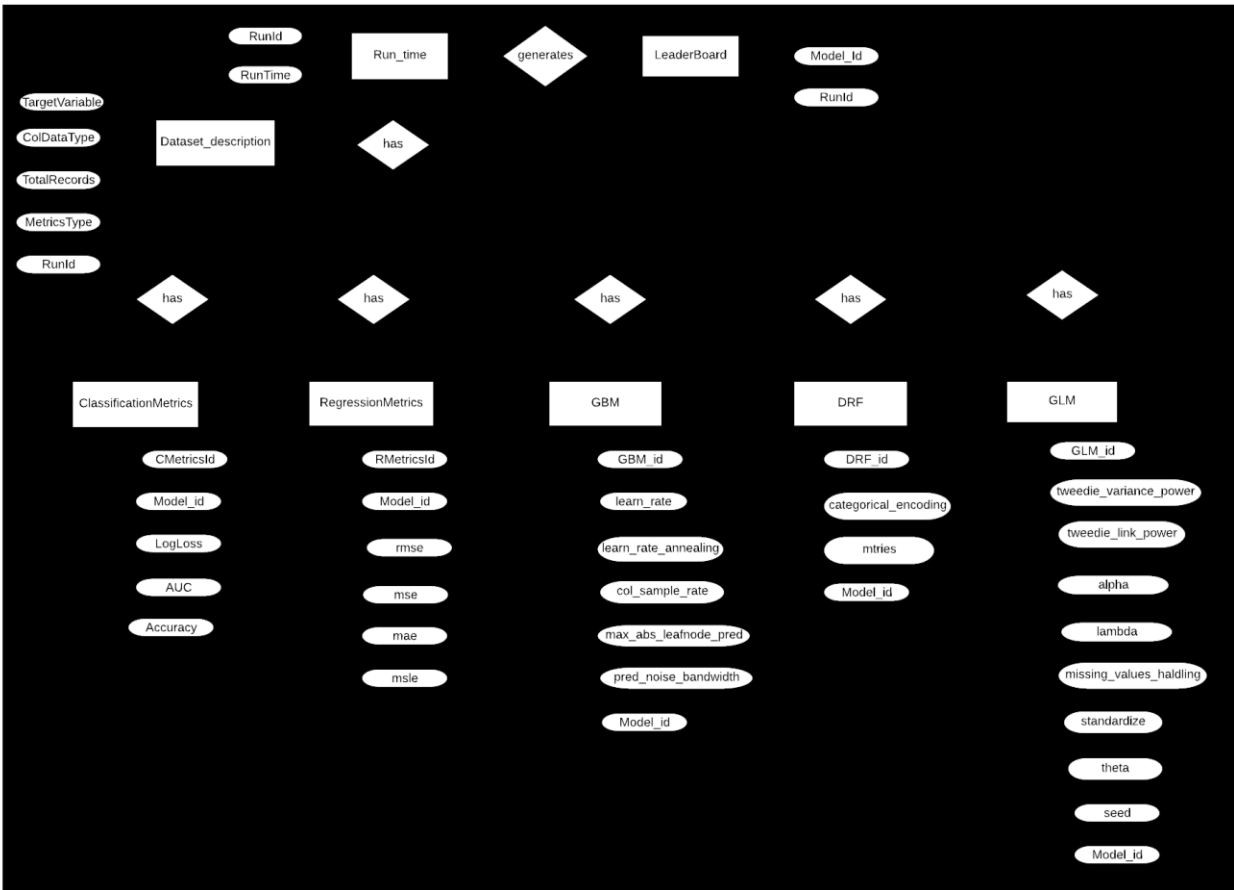
The objective of this project is to build a physical database storing details and values of the various hyperparameters generated through running the dataset into H2O. Finally, this database would help support a website which would help data enthusiasts get the best hyperparameter values for their respective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

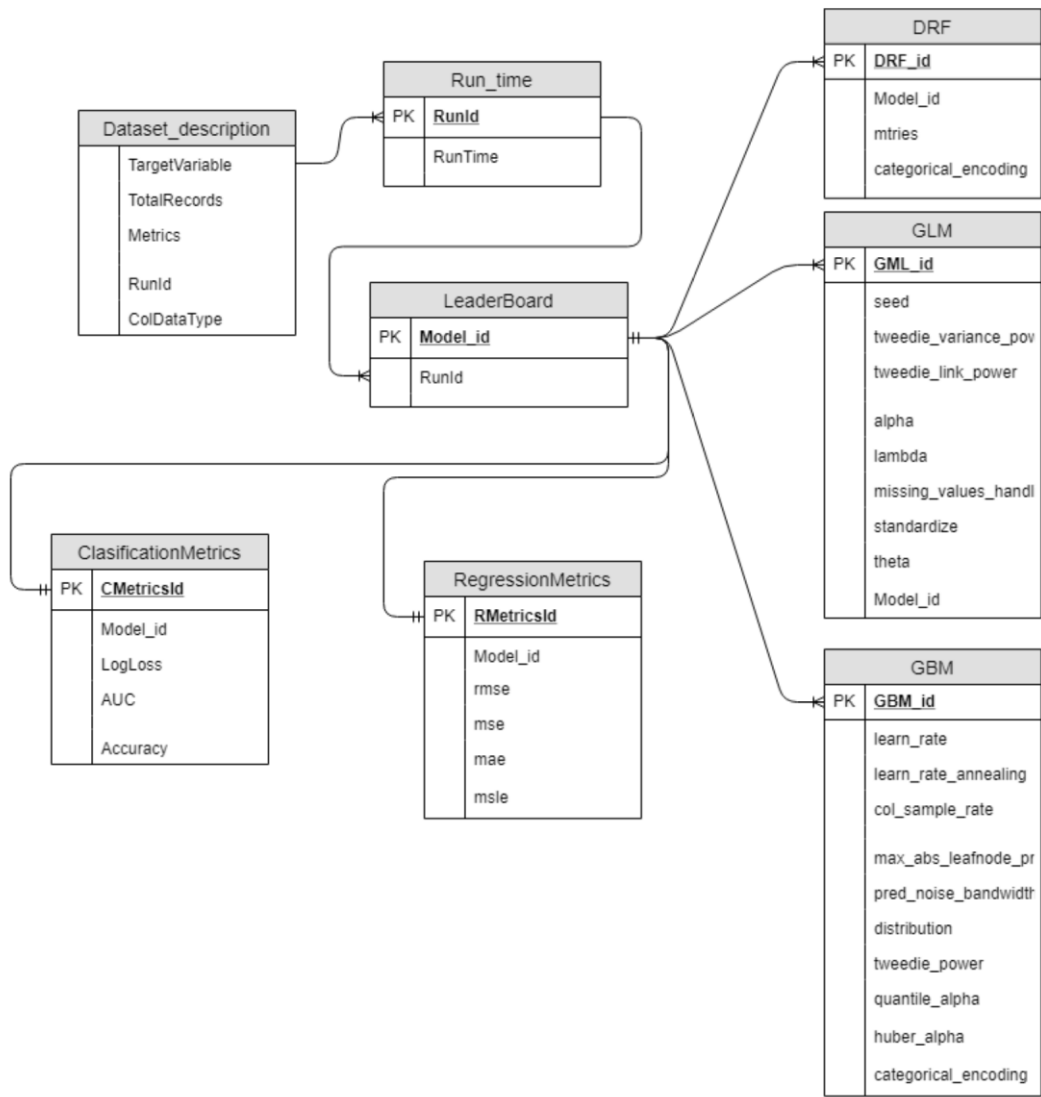
1. Conceptual Diagram
2. ER – Diagram
3. Normalization
4. Revised ER – Diagram
5. Converting JSON files into csv files based on the model
6. Creating the physical database
7. Writing practical use cases
8. Documentation and Professionalism

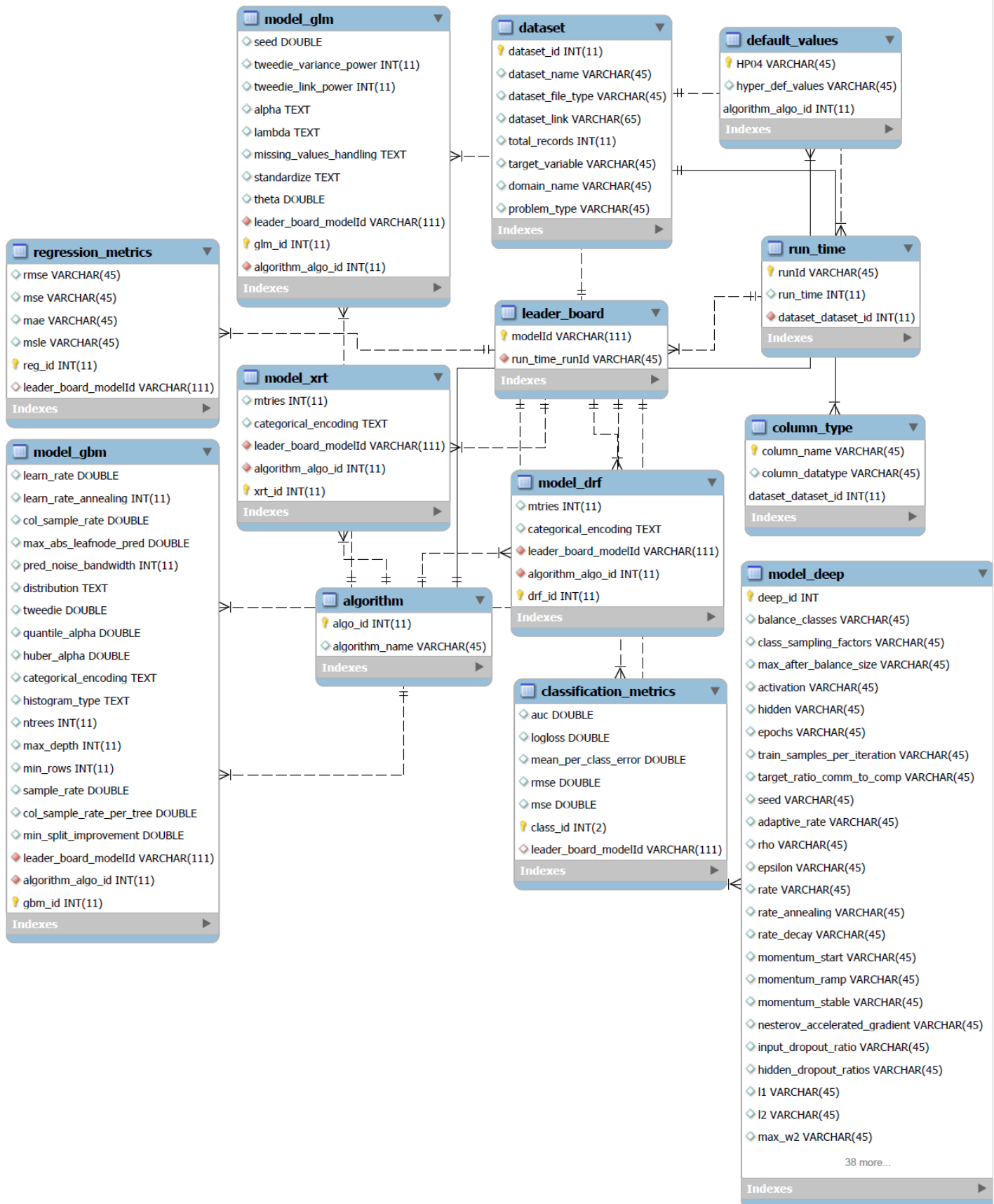
The objective for the loading the data in the physical database was to convert the JSON files into comma-separated value files (csv's). We did that using python. We then identified the entities, attributes, and relationships. Thus, we came up with the first version of our conceptual diagram.



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:





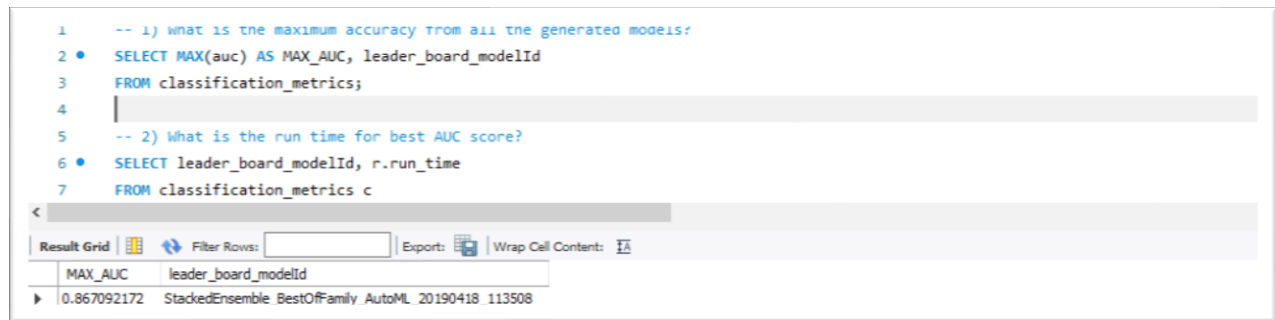
ERD final image

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:

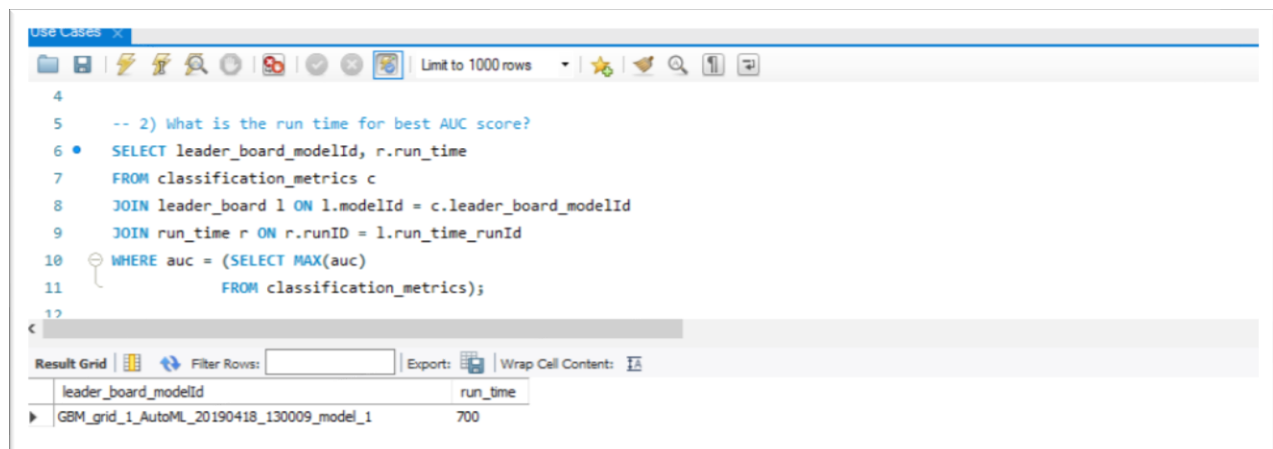
1) What is the maximum accuracy from all the generated models?

```
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);
```



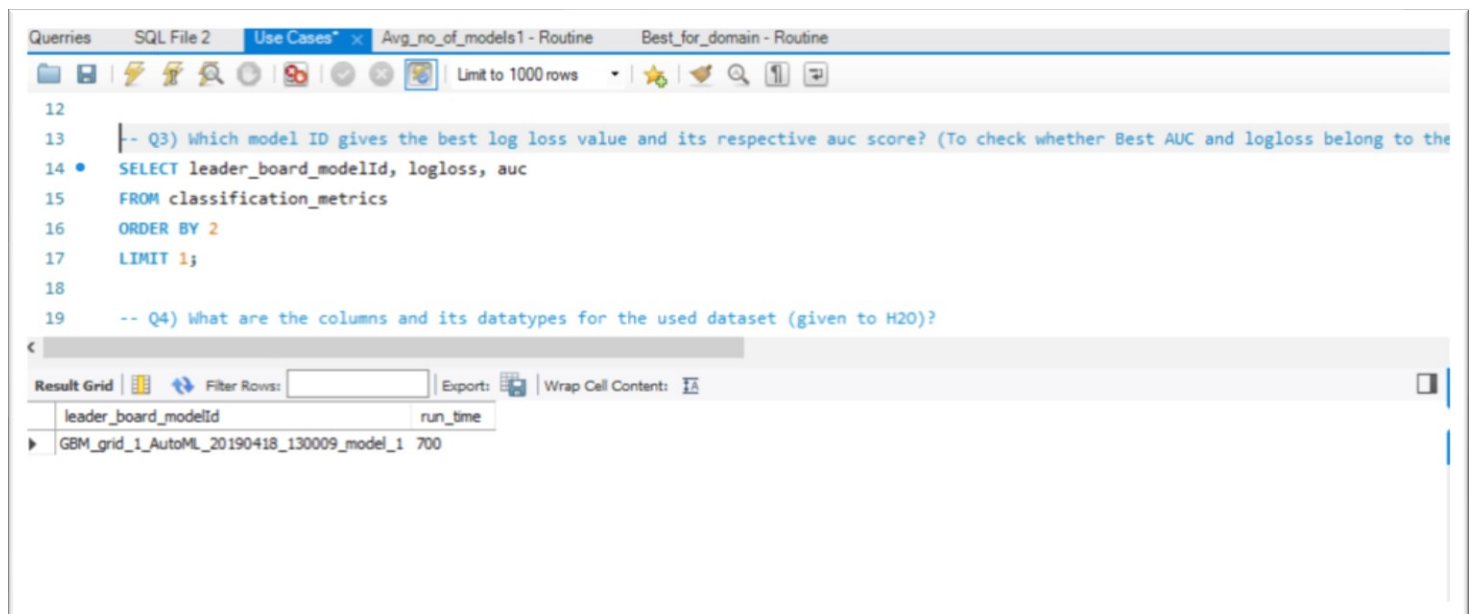
Q3) Which model ID gives the best log loss value? (To check whether Best AUC and logloss belong to the same model?)

```
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;
```

Use Cases

Limit to 1000 rows

```

16 ORDER BY 2
17 LIMIT 1;
18
19 -- Q4) What are the columns and its datatypes for the used dataset (given to H20)?
20 • SELECT column_name, column_datatype
21 FROM column_type
22 WHERE dataset_dataset_id = 1;
23

```

Result Grid

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);

Queries SQL File 2 Use Cases Avg_no_of_models1 - Routine Best_for_domain - Routine

Limit to 1000 rows

```

22 WHERE dataset_dataset_id = 1;
23
24 -- Q5) What is the best algorithm for classification type?
25 • SELECT LEFT(leader_board_modelId,3) AS Algo_Name
26 FROM classification_metrics c
27 JOIN leader_board l ON l.modelId = c.leader_board_modelId
28 WHERE auc = (SELECT MAX(auc)
29 FROM classification_metrics);

```

Result Grid

Algo_Name
GBM

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;
```

The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL code:

```
-- Q6) Get the max value for the max_depth hyperparameter
SELECT max_depth, leader_board_modelId
FROM model_gbm
ORDER BY 1 DESC
LIMIT 1;
```

The result grid shows the following data:

max_depth	leader_board_modelId
17	GBM_grid_1_AutoML_20190415_223833_model_8

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;
```

The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL code:

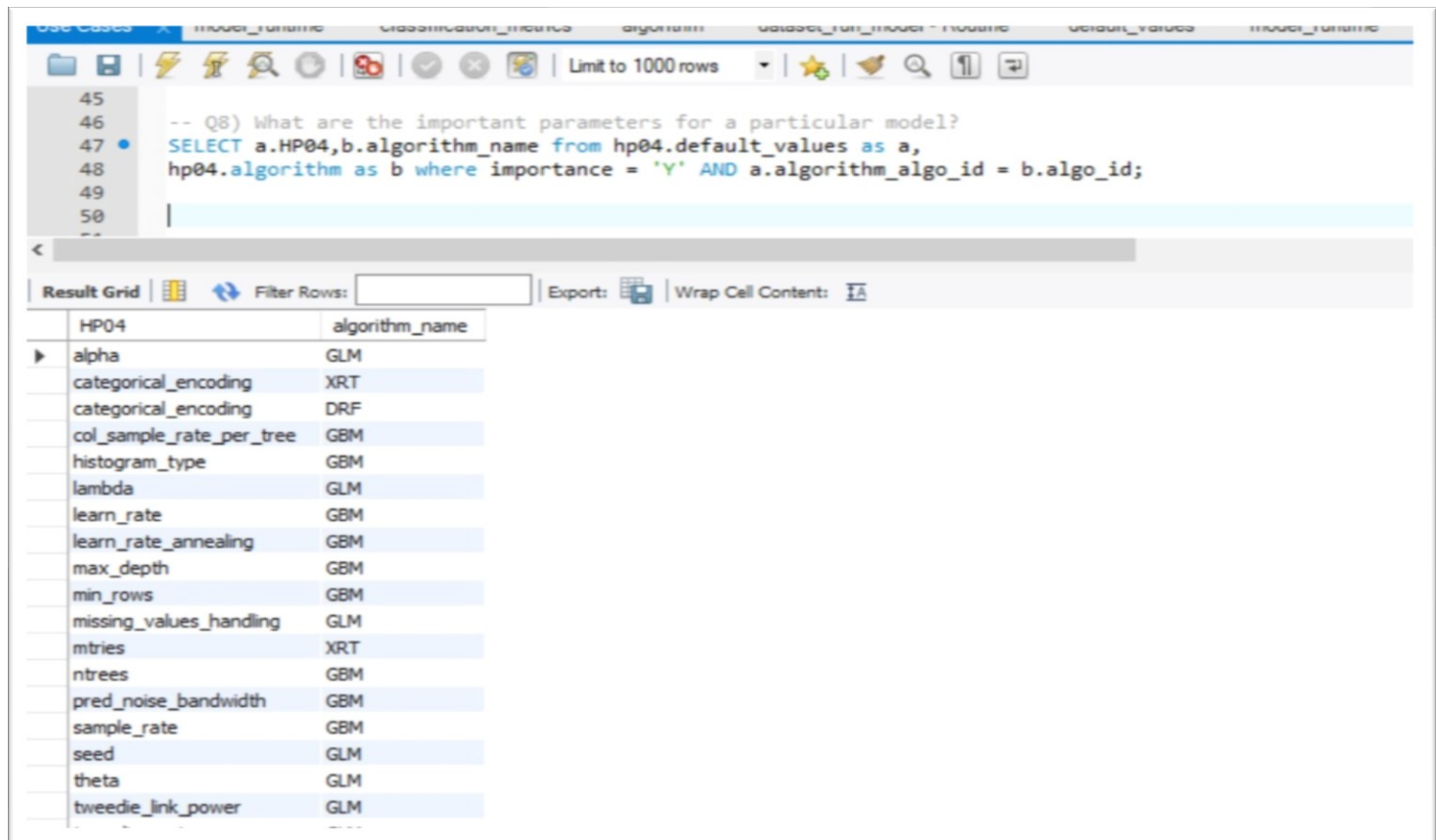
```
-- Q7) Common hypermeters from different algorithms
SELECT a.HP04,b.algorithm_name FROM hp04.default_values as a, hp04.algorithm as b
WHERE a.algorithm_algo_id = b.algo_id
GROUP BY a.HP04
HAVING COUNT(a.HP04) > 1;
```

The result grid shows the following data:

HP04	algorithm_name
categorical_encoding	GBM
distribution	GBM
missing_values_handling	GLM
mbtries	XRT
quantile_alpha	GBM
seed	GLM
standardize	GLM

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;
```

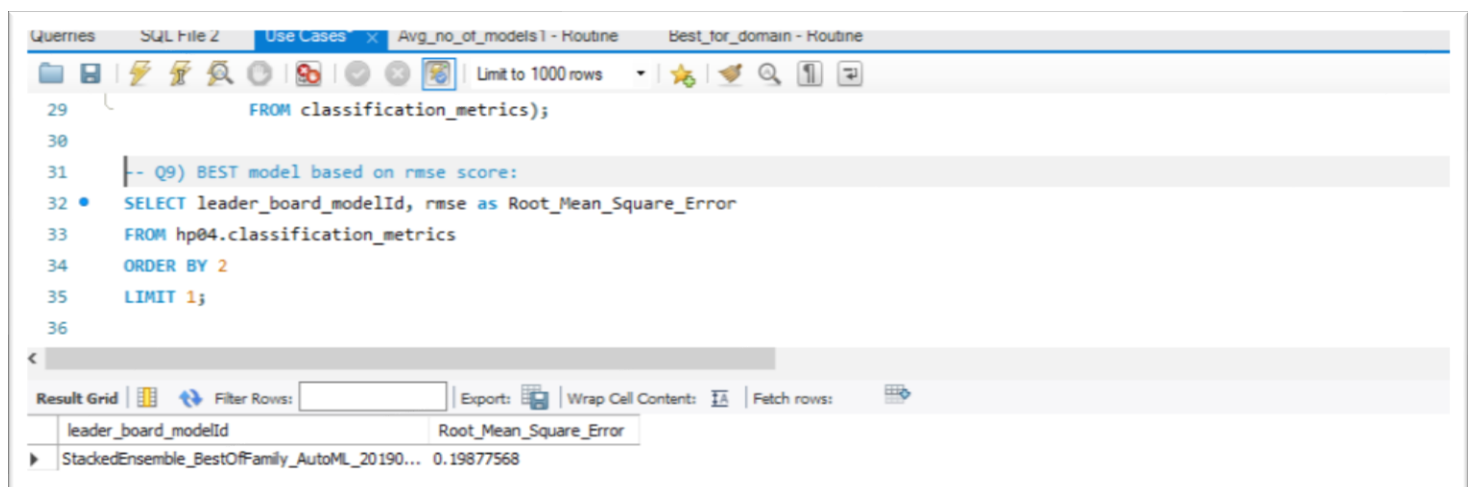


```
-- Q8) What are the important parameters for a particular model?  
SELECT a.HP04,b.algorithm_name from hp04.default_values as a,  
hp04.algorithm as b where importance = 'Y' AND a.algorithm_algo_id = b.algo_id;
```

HP04	algorithm_name
alpha	GLM
categorical_encoding	XRT
categorical_encoding	DRF
col_sample_rate_per_tree	GBM
histogram_type	GBM
lambda	GLM
learn_rate	GBM
learn_rate_annealing	GBM
max_depth	GBM
min_rows	GBM
missing_values_handling	GLM
mtries	XRT
ntrees	GBM
pred_noise_bandwidth	GBM
sample_rate	GBM
seed	GLM
theta	GLM
tweedie_link_power	GLM

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;
```

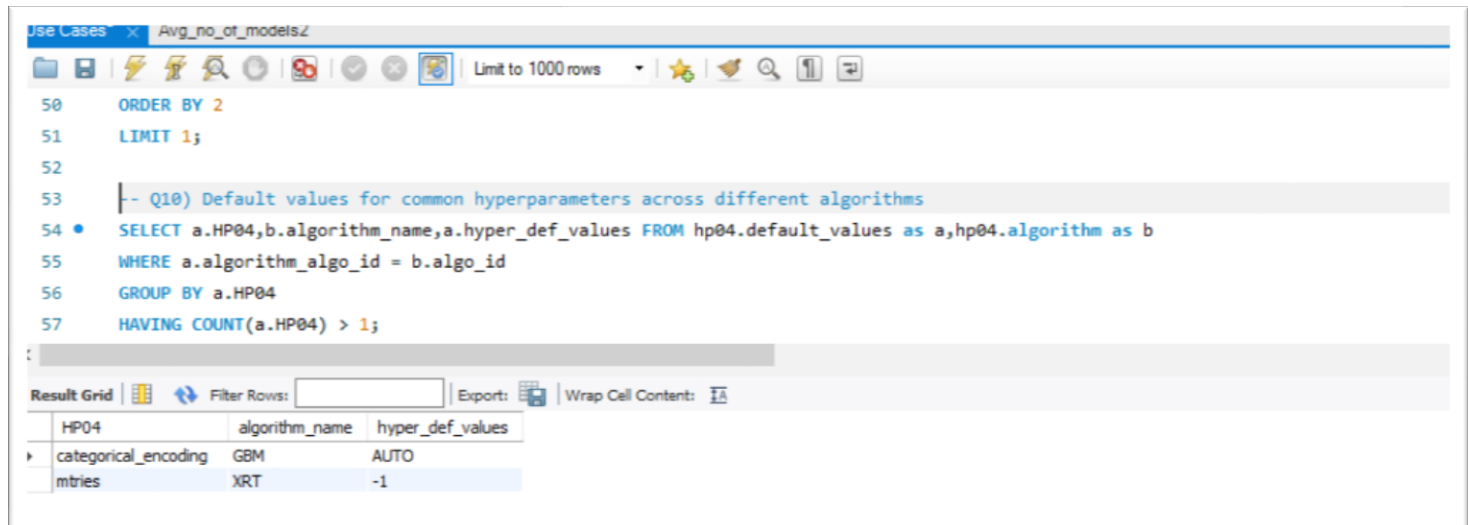


```
FROM classification_metrics);  
-- Q9) BEST model based on rmse score:  
SELECT leader_board_modelId, rmse as Root_Mean_Square_Error  
FROM hp04.classification_metrics  
ORDER BY 2  
LIMIT 1;
```

leader_board_modelId	Root_Mean_Square_Error
StackedEnsemble_BestOfFamily_AutoML_20190...	0.19877568

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;
```



The screenshot shows a SQL query editor window titled "Use Cases" with a file named "Avg_no_of_models.z". The query is as follows:

```
50 ORDER BY 2
51 LIMIT 1;
52
53 -- Q10) Default values for common hyperparameters across different algorithms
54 • SELECT a.HP04,b.algorithm_name,a.hyper_def_values FROM hp04.default_values as a, hp04.algorithm as b
55 WHERE a.algorithm_algo_id = b.algo_id
56 GROUP BY a.HP04
57 HAVING COUNT(a.HP04) > 1;
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows" input field, an "Export" button, and a "Wrap Cell Content" checkbox. The results are displayed in a table with three columns: "HP04", "algorithm_name", and "hyper_def_values".

HP04	algorithm_name	hyper_def_values
categorical_encoding	GBM	AUTO
mtries	XRT	-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;
```

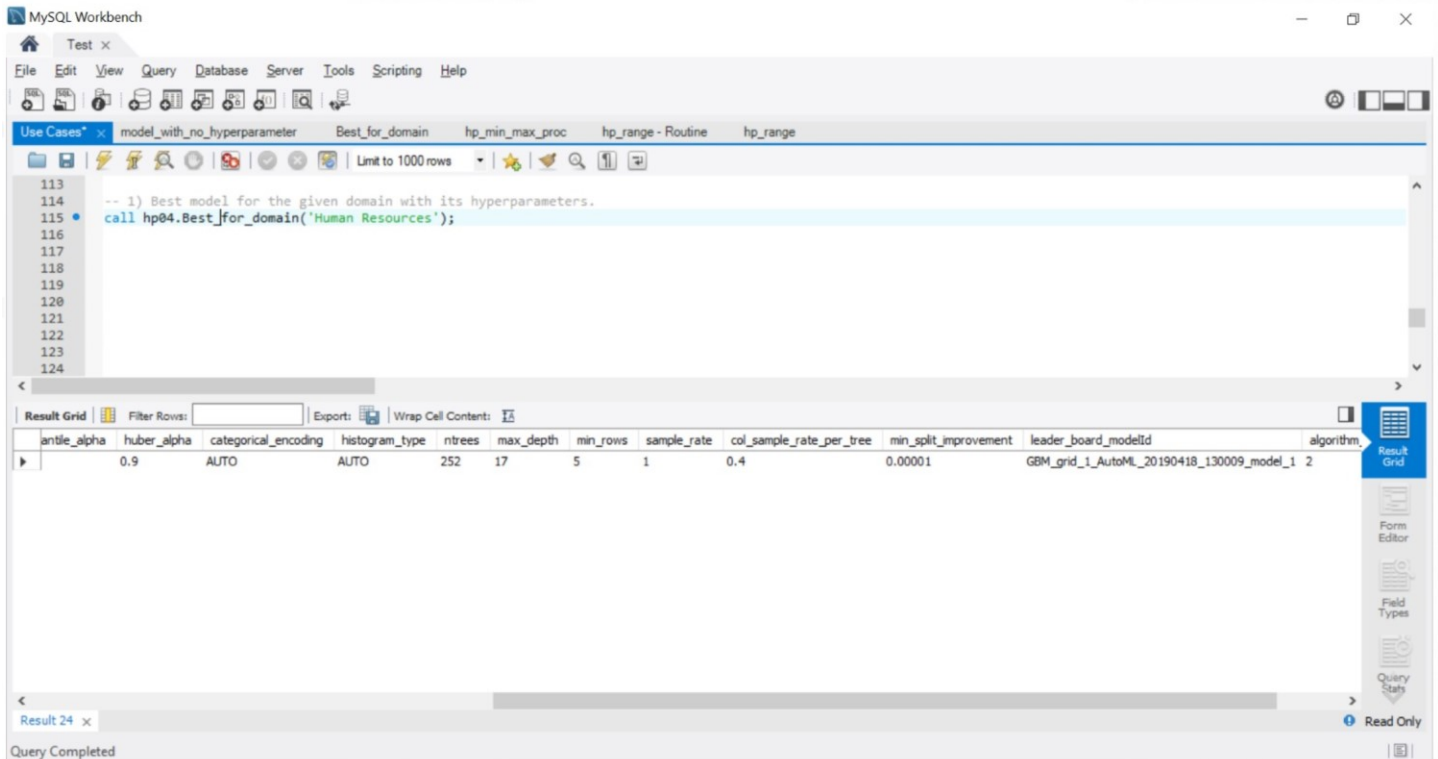
```
ELSEIF(LEFT(model,3) = 'XRT') THEN
```

```
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;
```

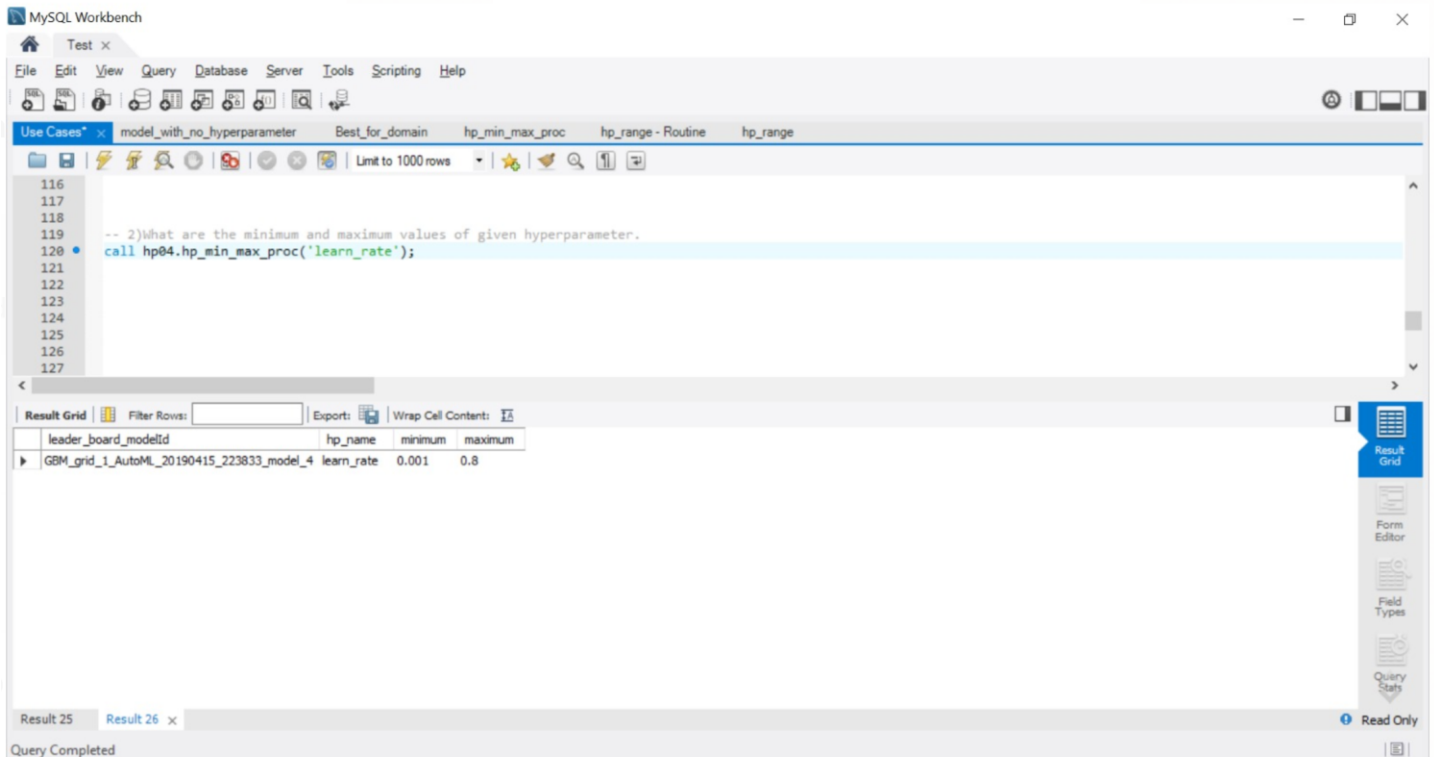
```
ELSEIF(parameter = 'ntrees') THEN
```

```
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

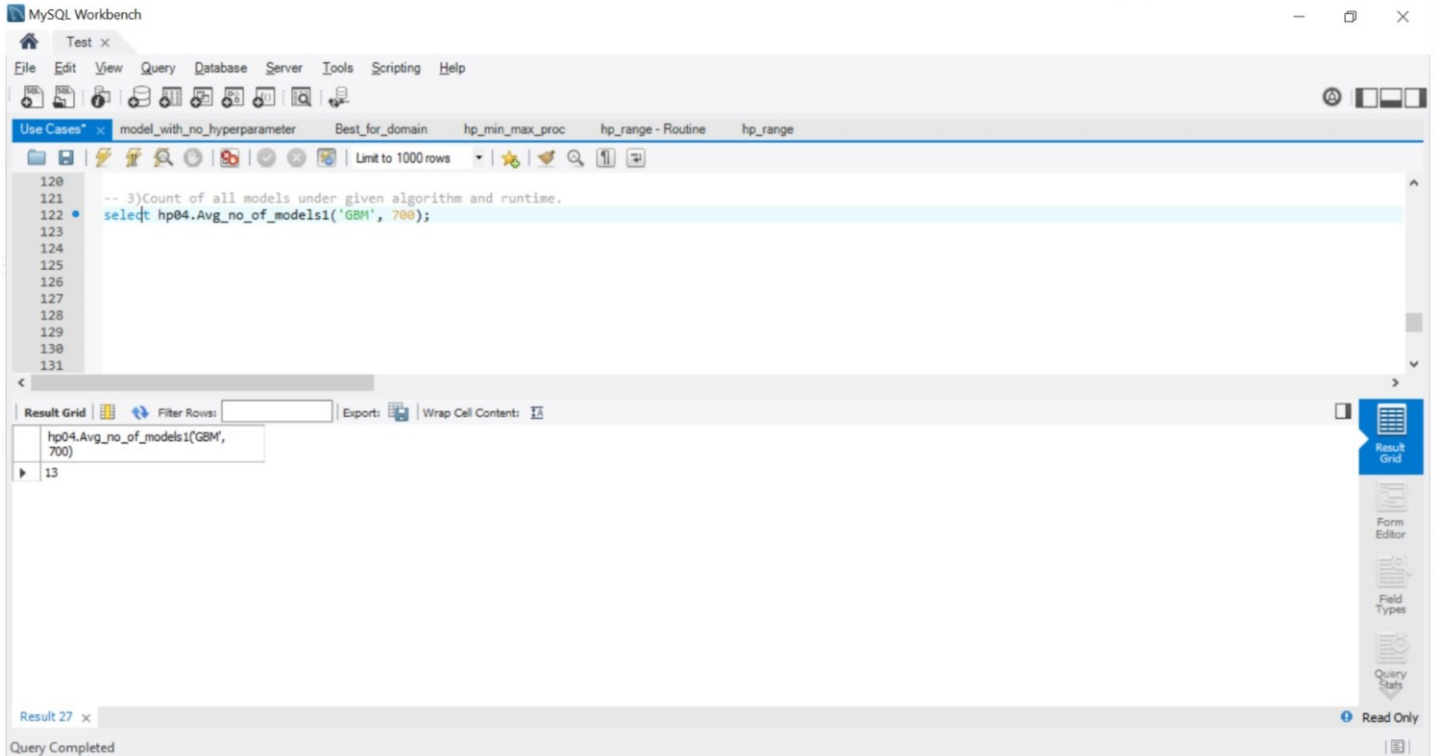
WHERE LEFT(modelId,3) LIKE concat(algorithm_name,'%')

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 ;
```

MySQL Workbench

Test x

File Edit View Query Database Server Tools Scripting Help

Use Cases* x model_with_no_hyperparameter Best_for_domain hp_min_max_proc hp_range - Routine hp_range

Limit to 1000 rows

```

120
121
122 -- 3)Count of all models under given algorithm and runtime.
123 • select hp04.Avg_no_of_models1('GBM', 700);
124
125 -- 4)What are the actual,default values and the range of learn_rate hyperparameter
126 • call hp04.hp_range();
127
128
129
130
131

```

Result Grid

learn_rate	default_value	hp_range
0.008	0.1	-0.092
0.1	0.1	0
0.1	0.1	0
0.005	0.1	-0.095
0.1	0.1	0
0.1	0.1	0
0.05	0.1	-0.05
0.1	0.1	0
0.001	0.1	-0.099
0.01	0.1	-0.090000000000000001
0.1	0.1	0
0.008	0.1	-0.092
0.005	0.1	-0.095
0.01	0.1	-0.090000000000000001

Result 28 x

Query Completed

Read Only

Views:

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

MySQL Workbench

Test x

File Edit View Query Database Server Tools Scripting Help

Use Cases* x model_with_no_hyperparameter Best_for_domain hp_min_max_proc hp_range - Routine hp_range

Limit to 1000 rows

```

68
69
70 -- 1) List of all datasets used
71
72 • CREATE VIEW `hp04`.`list_of_datasets` AS
73 select `hp04`.`dataset`.`dataset_id` AS `dataset_id`,`hp04`.`dataset`.`dataset_name` AS
74 `dataset_name`,`hp04`.`dataset`.`domain_name` AS `domain_name`,`hp04`.`dataset`.`problem_type`
75 AS `problem_type` from `hp04`.`dataset`;
76
77 • SELECT * FROM hp04.list_of_datasets;
78
79

```

Result Grid

dataset_id	dataset_name	domain_name	problem_type
1	Employee Access	Human Resources	C

list_of_datasets 20 x

SQL script saved to 'C:\Users\surya\Desktop\Python\PROJECT\Hyperparametr_DB04\Usecases\Use Cases.sql'

Read Only

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

MySQL Workbench interface showing a query to create a view for listing all runtimes and models. The query is as follows:

```
-- 2) List of all runtimes and respective models for those runtimes
CREATE VIEW `hp04`.`model_runtime` AS
SELECT a.modelId,b.run_time FROM leader_board AS a,run_time AS b WHERE a.run_time_runId = b.runId;
SELECT * FROM hp04.model_runtime;
```

The Result Grid shows the following data:

modelId	run_time
DRF_1_AutoML_20190418_115600	1000
GLM_grid_1_AutoML_20190418_115600_model_1	1000
StackedEnsemble_AllModels_AutoML_20190418_115600	1000
StackedEnsemble_BestOffFamily_AutoML_20190418_115600	1000
XGBoost_1_AutoML_20190418_115600	1000
XGBoost_2_AutoML_20190418_115600	1000
XRT_1_AutoML_20190418_115600	1000
DRF_1_AutoML_20190418_113508	500
GLM_grid_1_AutoML_20190418_113508_model_1	500
StackedEnsemble_AllModels_AutoML_20190418_113508	500
StackedEnsemble_BestOffFamily_AutoML_20190418_113508	500
XGBoost_1_AutoML_20190418_113508	500
XRT_1_AutoML_20190418_113508	500
Final winner: 1 & 2 (ML_20190418_120619)	1200

3) All hyperparameters and default values

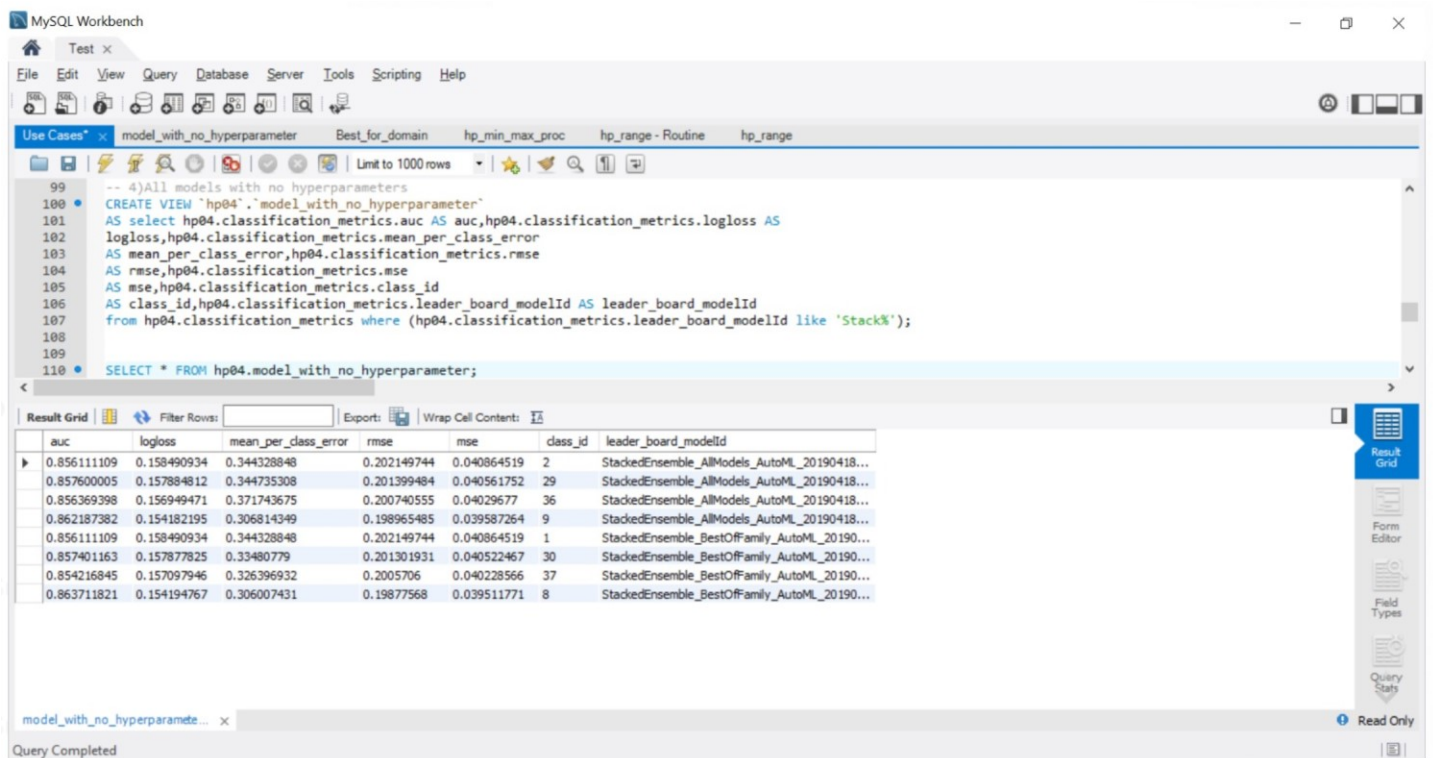
MySQL Workbench interface showing a query to create a view for listing all hyperparameters and default values. The query is as follows:

```
-- 3) All hyperparameters and default values
CREATE VIEW `hp04`.`default_hp_val` AS
select a.HP04 AS HP04,a.hyper_def_values AS hyper_def_values,b.algorithm_name AS algorithm_name
from (hp04.default_values a join hp04.algorithm b on((a.algorithm_algo_id = b.algo_id)));
SELECT * FROM hp04.default_hp_val;
```

The Result Grid shows the following data:

HP04	hyper_def_values	algorithm_name
alpha		GLM
lambda		GLM
missing_values_handling	MeanImputation	GLM
seed	-1	GLM
standardize	TRUE	GLM
theta	1.00E-10	GLM
tweedie_link_power	1	GLM
tweedie_variance_power	0	GLM
categorical_encoding	AUTO	GBM
col_sample_rate	1	GBM
col_sample_rate_per_tree	1	GBM
distribution	AUTO	GBM
histogram_type	AUTO	GBM
hp04.alpha	n a	GBM

4) All models with no hyperparameters



The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
-- 4) All models with no hyperparameters
CREATE VIEW `hp04`.`model_with_no_hyperparameter`
AS select hp04.classification_metrics.auc AS auc, hp04.classification_metrics.logloss AS
logloss, hp04.classification_metrics.mean_per_class_error
AS mean_per_class_error, hp04.classification_metrics.rmse
AS rmse, hp04.classification_metrics.mse
AS mse, hp04.classification_metrics.class_id
AS class_id, hp04.classification_metrics.leader_board_modelId AS leader_board_modelId
from hp04.classification_metrics where (hp04.classification_metrics.leader_board_modelId like 'Stack%');

SELECT * FROM hp04.model_with_no_hyperparameter;
```

The Results Grid shows the following data:

auc	logloss	mean_per_class_error	rmse	mse	class_id	leader_board_modelId
0.856111109	0.158490934	0.344328848	0.202149744	0.040864519	2	StackedEnsemble_AllModels_AutoML_20190418...
0.857600005	0.157884812	0.344735308	0.201399484	0.040561752	29	StackedEnsemble_AllModels_AutoML_20190418...
0.856369398	0.156949471	0.371743675	0.200740555	0.04029677	36	StackedEnsemble_AllModels_AutoML_20190418...
0.862187382	0.154182195	0.306814349	0.198965485	0.039587264	9	StackedEnsemble_AllModels_AutoML_20190418...
0.856111109	0.158490934	0.344328848	0.202149744	0.040864519	1	StackedEnsemble_BestOffFamily_AutoML_20190...
0.857401163	0.157877825	0.33480779	0.201301931	0.040522467	30	StackedEnsemble_BestOffFamily_AutoML_20190...
0.854216845	0.157097946	0.326396932	0.2005706	0.040228566	37	StackedEnsemble_BestOffFamily_AutoML_20190...
0.863711821	0.154194767	0.306007431	0.19877568	0.039511771	8	StackedEnsemble_BestOffFamily_AutoML_20190...

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> - H2O 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:

Copyright 2019 Tejas Munot, Manasa Vanga

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.