Northeastern University
College *of* Engineering

Northeastern University

Data Management and Database Design

INFO6210

Spring 2019

Hyperparameter Database Team 11

Project Report

By Darshan Durve & Shriram Karthikeyan

Darshan Durve
Shriram Karthikeyan

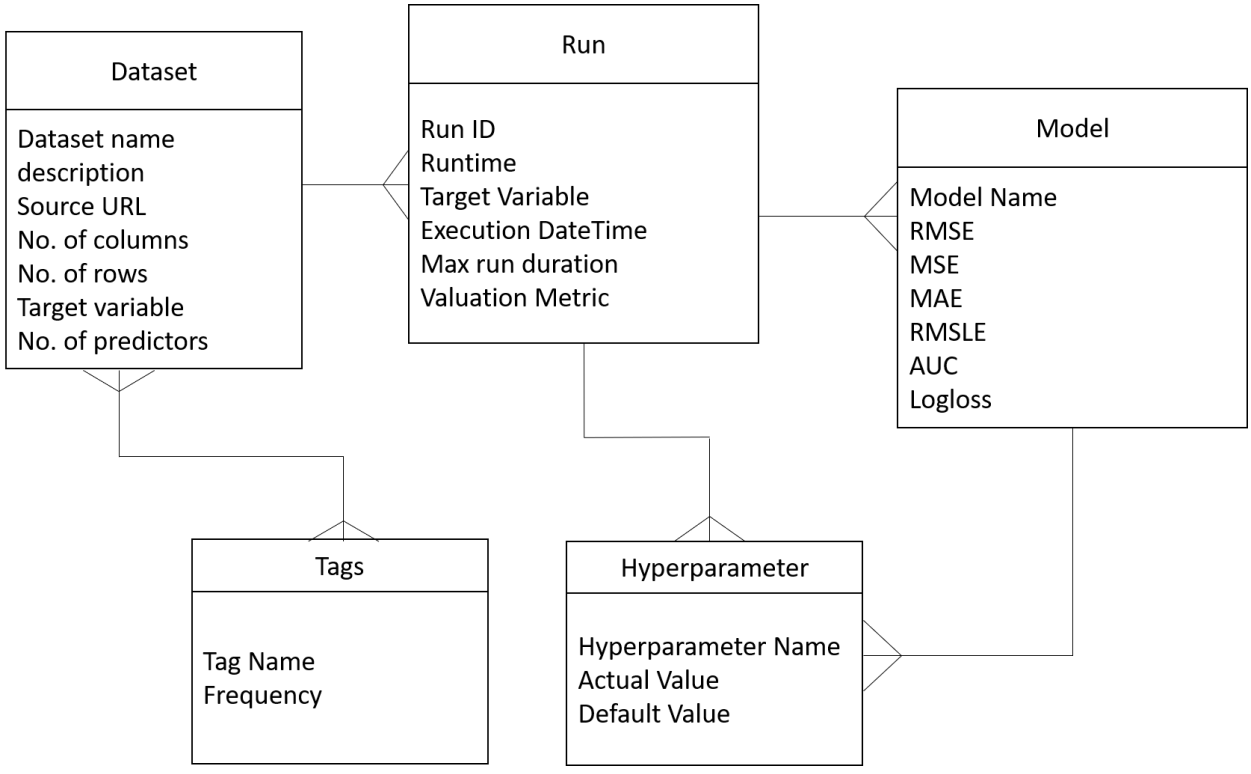
Abstract

The goal of this hyperparameter project is to create a database consisting of hyperparameters that were extracted by running various machine learning models (classification/regression) on multiple datasets. Using the database of hyperparameters that is generated we will suggest hyperparameters for the dataset that the user would like to run ML models on; this aspect of the project is not currently within our scope therefore we will limit ourselves to building a database of hyperparameters for one dataset.

The Dataset

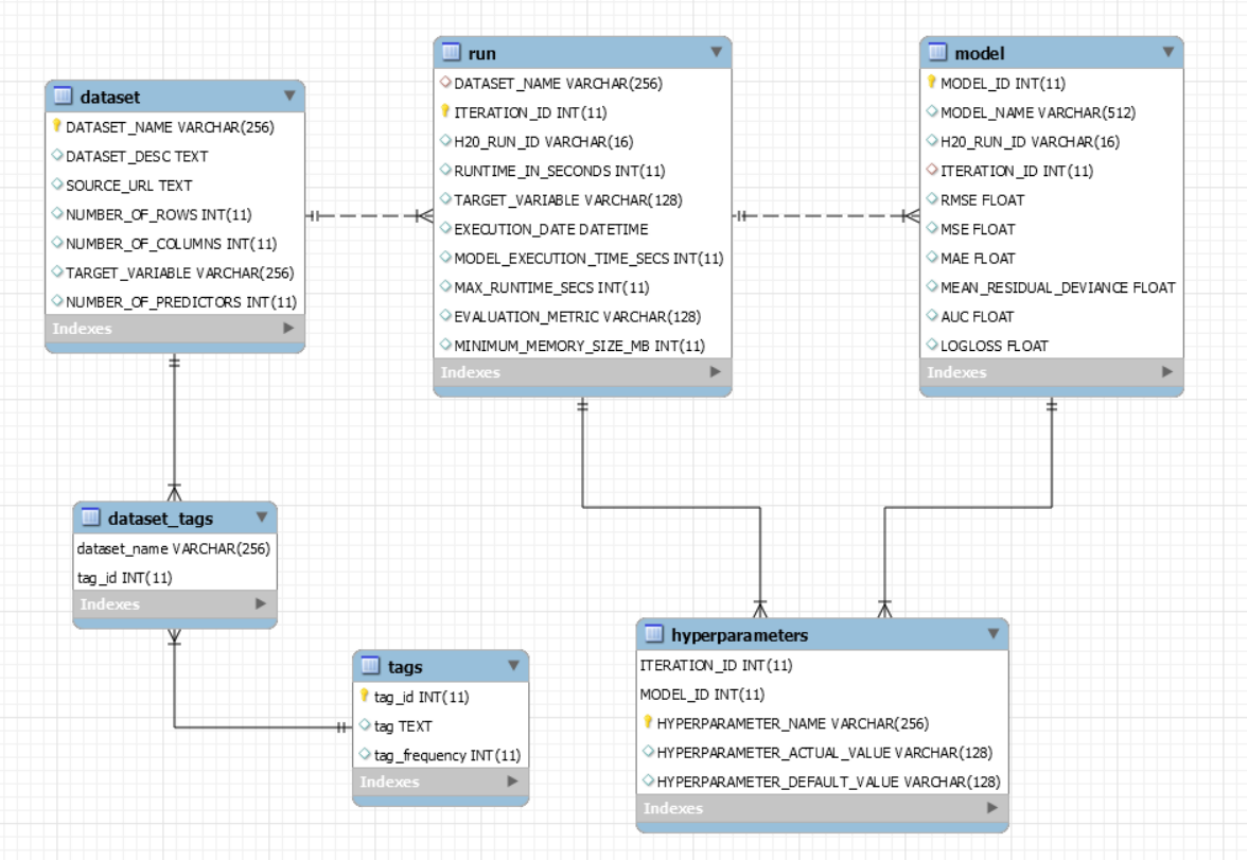
We have got the CalCOFI dataset from Kaggle.com. The data set represents the longest (1949-present) and most complete (more than 50,000 sampling stations) time series of oceanographic and larval fish data in the world. It includes abundance data on the larvae of over 250 species of fish; larval length frequency data and egg abundance data on key commercial species; and oceanographic and plankton data.

Conceptual Diagram



Darshan Durve
Shriram Karthikeyan

ER Diagram



Above E-R diagram explains the relationship between the models for each run with their hyperparameters. Each run has different models generated by H2O and each model has different hyperparameters associated to it. We have also stored tags to associate the dataset with keywords.

Normalization

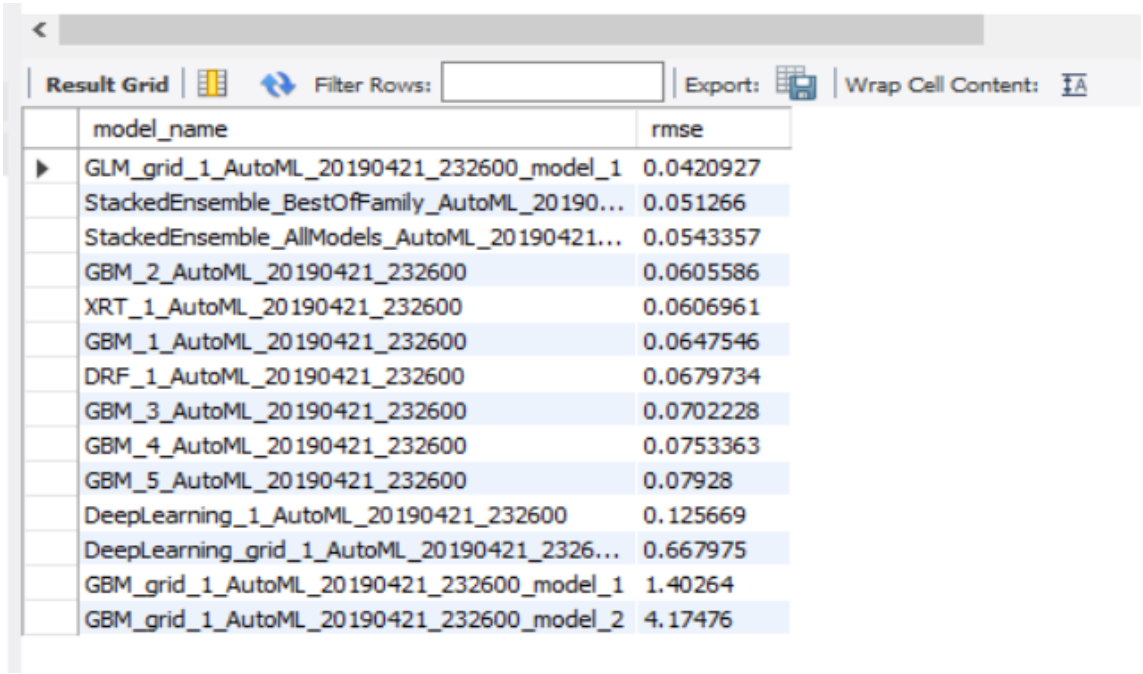
We have strived to follow 3NF in our physical data model and normalized table to a reasonable extent. For instance the hyperparameter table contains a composite key of Iteration_ID, Model_ID and the Hyperparameter_Name out of which Iteration_ID and Model_ID are foreign keys referencing the Run and Model tables respectively.

Darshan Durve
Shriram Karthikeyan

Use Cases

1. Retrieve the rmse value for the models of the lowest runtime.

- select m.model_name, m.rmse from model m join run r on m.iteration_id=r.iteration_id where r.max_runtime_secs=(select min(max_runtime_secs) from run);

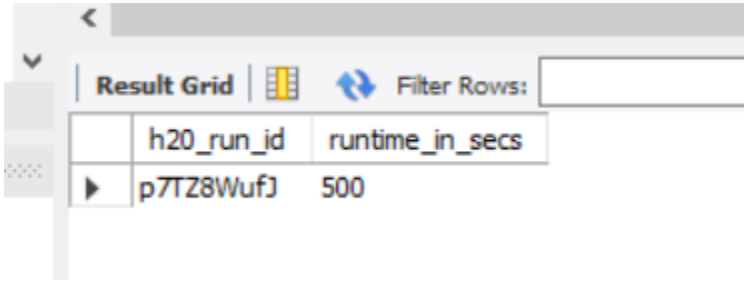


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

model_name	rmse
GLM_grid_1_AutoML_20190421_232600_model_1	0.0420927
StackedEnsemble_BestOfFamily_AutoML_20190...	0.051266
StackedEnsemble_AllModels_AutoML_20190421...	0.0543357
GBM_2_AutoML_20190421_232600	0.0605586
XRT_1_AutoML_20190421_232600	0.0606961
GBM_1_AutoML_20190421_232600	0.0647546
DRF_1_AutoML_20190421_232600	0.0679734
GBM_3_AutoML_20190421_232600	0.0702228
GBM_4_AutoML_20190421_232600	0.0753363
GBM_5_AutoML_20190421_232600	0.07928
DeepLearning_1_AutoML_20190421_232600	0.125669
DeepLearning_grid_1_AutoML_20190421_2326...	0.667975
GBM_grid_1_AutoML_20190421_232600_model_1	1.40264
GBM_grid_1_AutoML_20190421_232600_model_2	4.17476

2. Retrieve the runid from the meta-data for the lowest runtime.

- Select h20_run_id, max_runtime_secs as runtime_in_secs from run where max_runtime_secs=(select min(max_runtime_secs) from run);



Result Grid | Filter Rows:

h20_run_id	runtime_in_secs
p7TZ8WufJ	500

Darshan Durve
Shriram Karthikeyan

3. Retrieve the ntrees from the Random Forest algorithm.

- select m.model_name, h.hyperparameter_name, h.hyperparameter_actual_value
from model m join hyperparameters h on m.model_id=h.model_id
where m.model_name like 'drf%' and h.hyperparameter_name = 'ntrees' order by h.hyperparameter_actual_value desc;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

model_name	hyperparameter_name	hyperparameter_actual_value
DRF_1_AutoML_20190421_232600	ntrees	50
DRF_1_AutoML_20190421_233433	ntrees	50
DRF_1_AutoML_20190421_232600	ntrees	50
DRF_1_AutoML_20190421_233433	ntrees	50
DRF_1_AutoML_20190421_232600	ntrees	50
DRF_1_AutoML_20190421_232600	ntrees	50
DRF_1_AutoML_20190421_233433	ntrees	50
DRF_1_AutoML_20190421_232600	ntrees	50
DRF_1_AutoML_20190421_233433	ntrees	50
DRF_1_AutoML_20190422_001923	ntrees	43
DRF_1_AutoML_20190422_000108	ntrees	42
DRF_1_AutoML_20190422_000108	ntrees	42
DRF_1_AutoML_20190421_234613	ntrees	41
DRF_1_AutoML_20190421_234613	ntrees	41
DRF_1_AutoML_20190421_234613	ntrees	41

4. The default max_depth of the models

- select m.model_name, h.hyperparameter_name, h.hyperparameter_default_value from hyperparameters h join model m on h.model_id=m.model_id

where h.hyperparameter_name = 'max_depth';

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

model_name	hyperparameter_name	hyperparameter_default_value
GBM_4_AutoML_20190421_233433	max_depth	5
GBM_2_AutoML_20190421_232600	max_depth	5
XRT_1_AutoML_20190421_232600	max_depth	20
GBM_1_AutoML_20190421_233433	max_depth	5
GBM_2_AutoML_20190421_233433	max_depth	5

Result 58

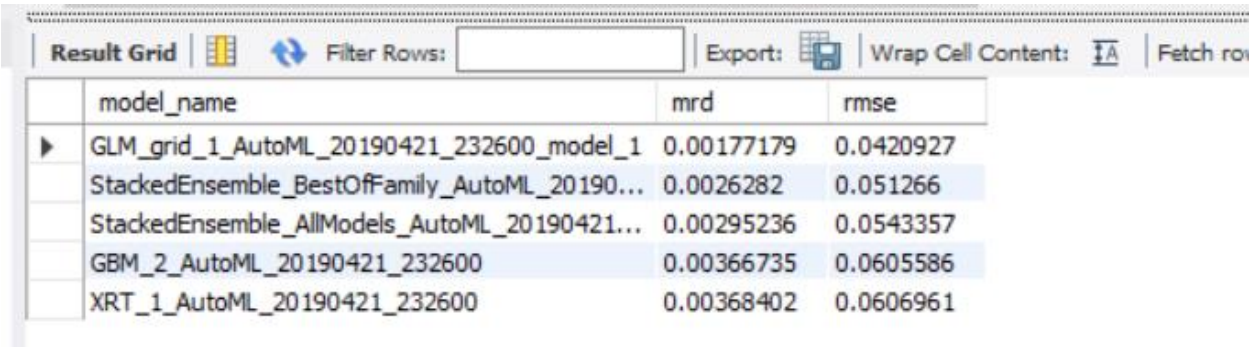
Darshan Durve
Shriram Karthikeyan

5. The mean residual deviance for the top 5 models of the lowest runtime.

#Based on rmse

```
select model_name, mean_residual_deviance as mrd, rmse from model m join run r on
m.iteration_id=r.iteration_id

where r.max_runtime_secs=(select min(max_runtime_secs) from run) order by rmse limit 5;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch ro

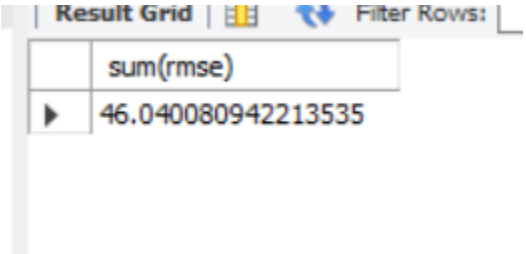
model_name	mrd	rmse
GLM_grid_1_AutoML_20190421_232600_model_1	0.00177179	0.0420927
StackedEnsemble_BestOfFamily_AutoML_20190...	0.0026282	0.051266
StackedEnsemble_AllModels_AutoML_20190421...	0.00295236	0.0543357
GBM_2_AutoML_20190421_232600	0.00366735	0.0605586
XRT_1_AutoML_20190421_232600	0.00368402	0.0606961

6. Sum of rmse values of any least or highest runtime.

#For highest

```
select sum(rmse) from model m join run r on m.iteration_id=r.iteration_id

where r.max_runtime_secs in (select max(max_runtime_secs) from run);
```



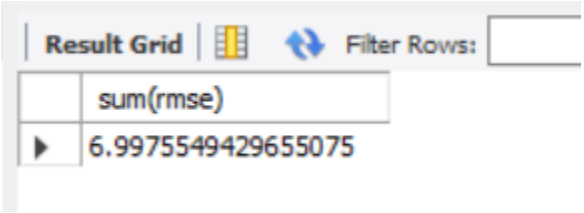
Result Grid | Filter Rows:

sum(rmse)
46.040080942213535

#For lowest

```
select sum(rmse) from model m join run r on m.iteration_id=r.iteration_id

where r.max_runtime_secs in (select min(max_runtime_secs) from run);
```



Result Grid | Filter Rows:

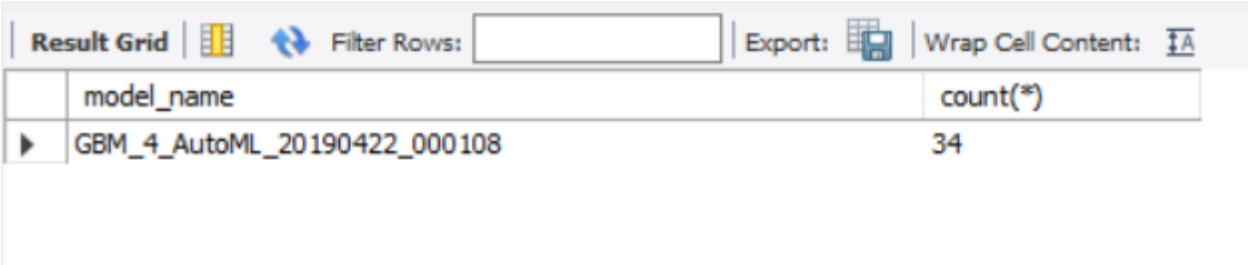
sum(rmse)
6.9975549429655075

Darshan Durve
Shriram Karthikeyan

7. The number of models of the same algorithm for a particular runtime(1100).

select model_name, count(*) from model m join run r on m.iteration_id=r.iteration_id

where r.max_runtime_secs = '1100' and model_name like 'GBM%';



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

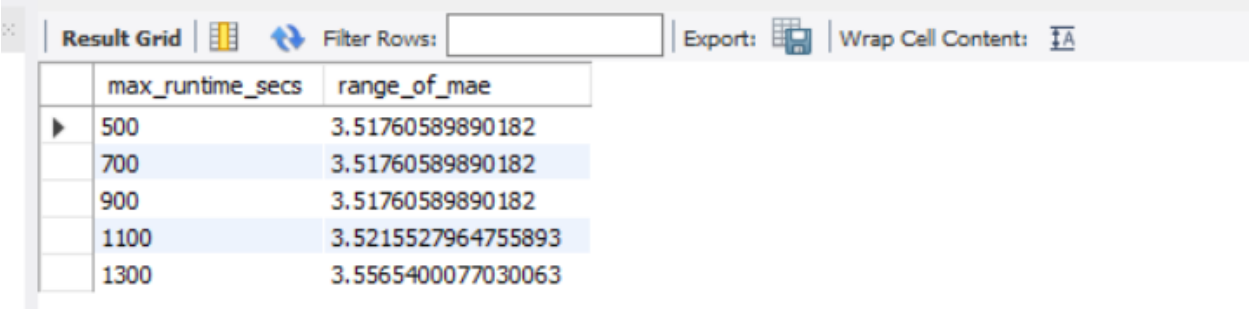
model_name	count(*)
GBM_4_AutoML_20190422_000108	34

8. The difference between the highest and lowest mae values of the models for a runtime.

- select r.max_runtime_secs, max(m.mae)-min(m.mae) as range_of_mae

from run r join model m on m.iteration_id=r.iteration_id group by r.max_runtime_secs

order by max_runtime_secs;

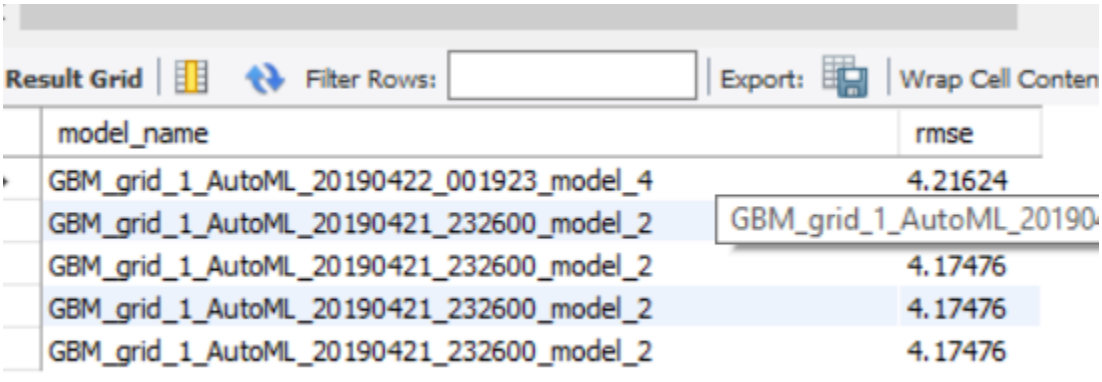


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

max_runtime_secs	range_of_mae
500	3.51760589890182
700	3.51760589890182
900	3.51760589890182
1100	3.5215527964755893
1300	3.5565400077030063

9. List of top 5 models with low performance based on their rmse values

- select model_name, rmse from model order by rmse desc limit 5;



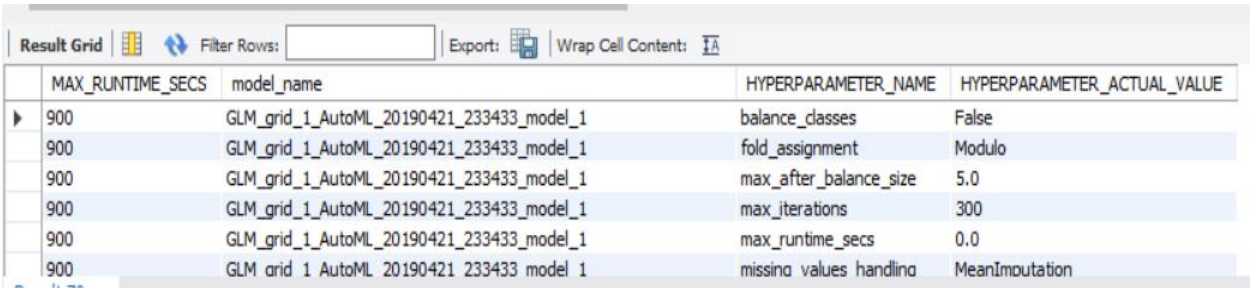
Result Grid | Filter Rows: | Export: | Wrap Cell Conten

model_name	rmse
GBM_grid_1_AutoML_20190422_001923_model_4	4.21624
GBM_grid_1_AutoML_20190421_232600_model_2	GBM_grid_1_AutoML_20190
GBM_grid_1_AutoML_20190421_232600_model_2	4.17476
GBM_grid_1_AutoML_20190421_232600_model_2	4.17476
GBM_grid_1_AutoML_20190421_232600_model_2	4.17476

Darshan Durve
Shriram Karthikeyan

10. Suggest me a list of run times, model_names, hyperparameters and their values to get the lowest rmse

- select r.MAX_RUNTIME_SECS, m.model_name, h.HYPERPARAMETER_NAME, h.HYPERPARAMETER_ACTUAL_VALUE
  from run r, model m, hyperparameters h
  where r.iteration_id=m.iteration_id and m.model_id=h.model_id
  and m.rmse = (select min(rmse) from model);



MAX_RUNTIME_SECS	model_name	HYPERPARAMETER_NAME	HYPERPARAMETER_ACTUAL_VALUE
900	GLM_grid_1_AutoML_20190421_233433_model_1	balance_classes	False
900	GLM_grid_1_AutoML_20190421_233433_model_1	fold_assignment	Modulo
900	GLM_grid_1_AutoML_20190421_233433_model_1	max_after_balance_size	5.0
900	GLM_grid_1_AutoML_20190421_233433_model_1	max_iterations	300
900	GLM_grid_1_AutoML_20190421_233433_model_1	max_runtime_secs	0.0
900	GLM_grid_1_AutoML_20190421_233433_model_1	missing_values_handling	MeanImputation

Darshan Durve
Shriram Karthikeyan

Functions

1. Suggest a dataset based on the given tag –

CREATE DEFINER=`root`@`localhost` FUNCTION `fetch_dataset`(tag_name varchar(128)) RETURNS varchar(128) CHARSET utf8mb4

BEGIN

declare dataset varchar(128);

select distinct d.dataset_name into dataset from tags t join dataset_tags d on t.tag_id=d.tag_id where tag like concat('%',tag_name,'%');

RETURN dataset;

END

```
6 •   select fetch_dataset('ocean') as dataset;
```

Result Grid | Filter Rows: | Export:

dataset
CalCOFI - bottle.csv

2. Suggest best value of specified hyperparameter of specified model at specified runtime

CREATE DEFINER=`root`@`localhost` FUNCTION `best_hyperparameter`(runtime varchar(128), model varchar(128), hyper varchar(128)) RETURNS varchar(128) CHARSET utf8mb4

BEGIN

declare best_value varchar(128);

select distinct HYPERPARAMETER_ACTUAL_VALUE into best_value from model m join run r on m.iteration_id=r.iteration_id

join hyperparameters h on m.model_id=h.model_id

where r.max_runtime_secs = runtime and

m.model_name like concat('%',model,'%')

and h.hyperparameter_name like concat('%',hyper,'%')

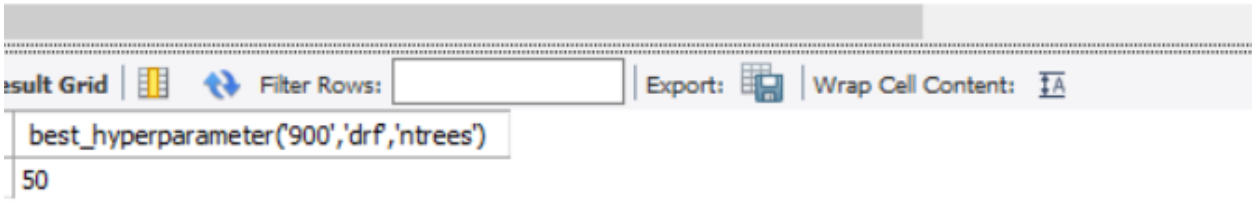
and m.rmse = (select min(m.rmse) from model m join run r on m.iteration_id=r.iteration_id

where r.max_runtime_secs = runtime and m.model_name like concat('%',model,'%'));

```
RETURN best_value;

END
```

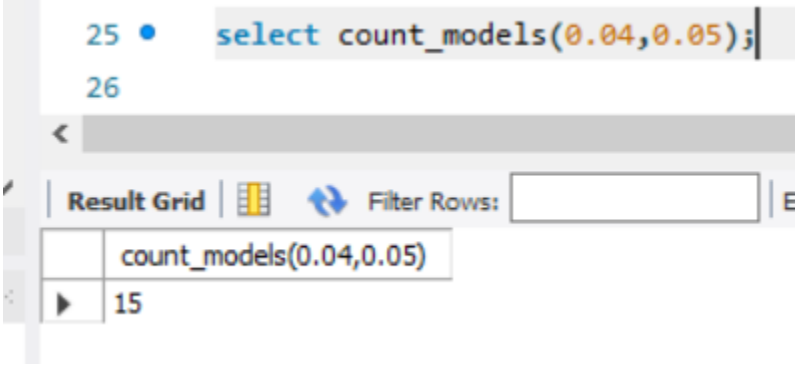
```
select best_hyperparameter('900','drf','ntrees');
select best_hyperparameter('700','gbm','fold_assignment');
```



3. Get number of models for a given range of RMSE values

```
CREATE DEFINER=`root`@`localhost` FUNCTION `count_models`(FROM_RMSE float, TO_RMSE float)
RETURNS int(11)
   DETERMINISTIC
BEGIN
DECLARE  MODEL_COUNT int;
select count(model_name) into model_count from model where rmse between from_rmse and
to_rmse;
RETURN MODEL_COUNT;
END
```

```
select count_models(0.04,0.05);
```



4. Get the best model for a given runtime

```
CREATE DEFINER=`root`@`localhost` FUNCTION `best_model`(runtime int) RETURNS varchar(128)
CHARSET utf8mb4
   DETERMINISTIC
BEGIN
DECLARE model varchar(128);

select model_name into model from model m join run r on m.iteration_id=r.iteration_id
where max_runtime_secs = 900
group by m.model_name, rmse
order by rmse desc limit 1;

RETURN model;
END
```

Darshan Durve
Shriram Karthikeyan

```
15 •    select best_model(700);
16
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

best_model(700)
GBM_grid_1_AutoML_20190421_232600_model_2

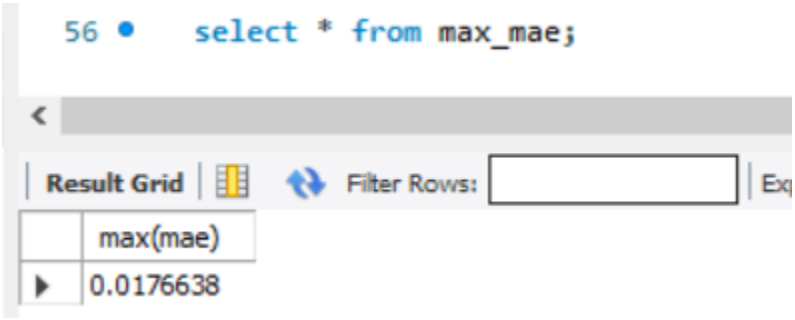
Darshan Durve
Shriram Karthikeyan

Views

View 1

1) Create a view to get the highest rmse value for the drf algorithm with the least runtime

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `hyperparameter_db_11`.`max_mae` AS
    SELECT
        MAX(`m`.`MAE`) AS `max(mae)`
    FROM
        ((`hyperparameter_db_11`.`model` `m`
        JOIN `hyperparameter_db_11`.`hyperparameters` `h`)
        JOIN `hyperparameter_db_11`.`run` `r`)
    WHERE
        ((`h`.`MODEL_ID` = `m`.`MODEL_ID`)
            AND (`m`.`ITERATION_ID` = `r`.`ITERATION_ID`)
            AND (`m`.`MODEL_NAME` LIKE '%drf%')
            AND (`r`.`MAX_RUNTIME_SECS` = (SELECT
                MIN(`hyperparameter_db_11`.`run`.`MAX_RUNTIME_SECS`)
            FROM
                `hyperparameter_db_11`.`run`)))
```



Darshan Durve
Shriram Karthikeyan

View 2

2) Create a view to display max and min values of Tweedie power for all deep learning algorithms for any all runtime

- CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
  VIEW `hyperparameter_db_11`.`tweedie_power` AS
  SELECT
  `r`.`MAX_RUNTIME_SECS` AS `max_runtime_secs`,
  `m`.`MODEL_NAME` AS `model_name`,
  `h`.`HYPERPARAMETER_NAME` AS `hyperparameter_name`,
  MAX(`h`.`HYPERPARAMETER_ACTUAL_VALUE`) AS
  `max(h.HYPERPARAMETER_ACTUAL_VALUE)`,
  MIN(`h`.`HYPERPARAMETER_ACTUAL_VALUE`) AS
  `min(h.HYPERPARAMETER_ACTUAL_VALUE)`
  FROM
  ((`hyperparameter_db_11`.`model` `m`
  JOIN `hyperparameter_db_11`.`hyperparameters` `h`)
  JOIN `hyperparameter_db_11`.`run` `r`)
  WHERE
  ((`h`.`MODEL_ID` = `m`.`MODEL_ID`)
  AND (`m`.`ITERATION_ID` = `r`.`ITERATION_ID`)
  AND (`h`.`HYPERPARAMETER_NAME` = 'tweedie_power')
  AND (`m`.`MODEL_NAME` LIKE '%deep%'))
  GROUP BY `r`.`MAX_RUNTIME_SECS`

```
67 •  select * from tweedie_power;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

max_runtime_secs	model_name	hyperparameter_name	max(h.HYPERPARAMETER_ACTUAL_VALUE)	min(h.HYPERPARAMETER_ACTUAL_VALUE)
900	DeepLearning_1_AutoML_201904...	tweedie_power	1.5	1.5
1100	DeepLearning_1_AutoML_201904...	tweedie_power	1.5	1.5
500	DeepLearning_1_AutoML_201904...	tweedie_power	1.5	1.5
700	DeepLearning_1_AutoML_201904...	tweedie_power	1.5	1.5
1300	DeepLearning_1_AutoML_201904...	tweedie_power	1.5	1.5

View 3

View that contains the best hyperparameters by dataset

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW `best_hp_by_dataset` AS
```

```
SELECT
    `d`.`DATASET_NAME` AS `dataset_name`,
    `h`.`HYPERPARAMETER_NAME` AS `HYPERPARAMETER_NAME`,
    `h`.`HYPERPARAMETER_ACTUAL_VALUE` AS `HYPERPARAMETER_ACTUAL_VALUE`,
    `h`.`HYPERPARAMETER_DEFAULT_VALUE` AS `HYPERPARAMETER_DEFAULT_VALUE`
FROM
    (((`dataset` `d`
    LEFT JOIN `run` `r` ON ((`d`.`DATASET_NAME` = `r`.`DATASET_NAME`)))
    LEFT JOIN `model` `m` ON ((`r`.`ITERATION_ID` = `m`.`MODEL_ID`)))
    LEFT JOIN `hyperparameters` `h` ON (((`m`.`MODEL_ID` = `h`.`MODEL_ID`)
        AND (`m`.`ITERATION_ID` = `h`.`ITERATION_ID`))))
WHERE
    (`m`.`RMSE` = (SELECT
            MIN(`m2`.`RMSE`)
        FROM
            ((`dataset` `d2`
            LEFT JOIN `run` `r2` ON ((`d2`.`DATASET_NAME` = `r2`.`DATASET_NAME`)))
            LEFT JOIN `model` `m2` ON ((`r2`.`ITERATION_ID` = `m2`.`MODEL_ID`)))
        WHERE
            (`d2`.`DATASET_NAME` = `d`.`DATASET_NAME`)))
```

```
25 •    select * from best_hp_by_dataset;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

dataset_name	HYPERPARAMETER_NAME	HYPERPARAMETER_ACTUAL_VALUE	HYPERPARAMETER_DEFAULT_VALUE
CalCOFI - bottle.csv	balance_classes	False	False
CalCOFI - bottle.csv	fold_assignment	Modulo	AUTO
CalCOFI - bottle.csv	max_after_balance_size	5.0	5.0
CalCOFI - bottle.csv	max_iterations	300	-1
CalCOFI - bottle.csv	max_runtime_secs	0.0	0.0
CalCOFI - bottle.csv	missing_values_handling	MeanImputation	MeanImputation
CalCOFI - bottle.csv	seed	6238141046563332l4	-1
CalCOFI - bottle.csv	standardize	True	True

This view contains the best set of hyperparameters for all datasets which makes it easier for the end user to query without having to join tables and see only the necessary information.

View 4

This view gives the summary count of the number of runs, models and hyperparameters by dataset

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `dataset_summary` AS
    SELECT
        `d`.`DATASET_NAME` AS `dataset_name`,
        COUNT(DISTINCT `r`.`ITERATION_ID`) AS `number_of runs`,
        COUNT(DISTINCT `m`.`MODEL_ID`) AS `number_of_models`,
        COUNT(DISTINCT `h`.`HYPERPARAMETER_NAME`) AS `number_of_hyperparameters`
```

```
FROM
    (((`dataset` `d`
    LEFT JOIN `run` `r` ON ((`d`.`DATASET_NAME` = `r`.`DATASET_NAME`)))
    LEFT JOIN `model` `m` ON ((`r`.`ITERATION_ID` = `m`.`MODEL_ID`)))
    LEFT JOIN `hyperparameters` `h` ON (((`m`.`MODEL_ID` = `h`.`MODEL_ID`)
        AND (`m`.`ITERATION_ID` = `h`.`ITERATION_ID`))))
```

```
21 •  select * from dataset_summary;
22
```

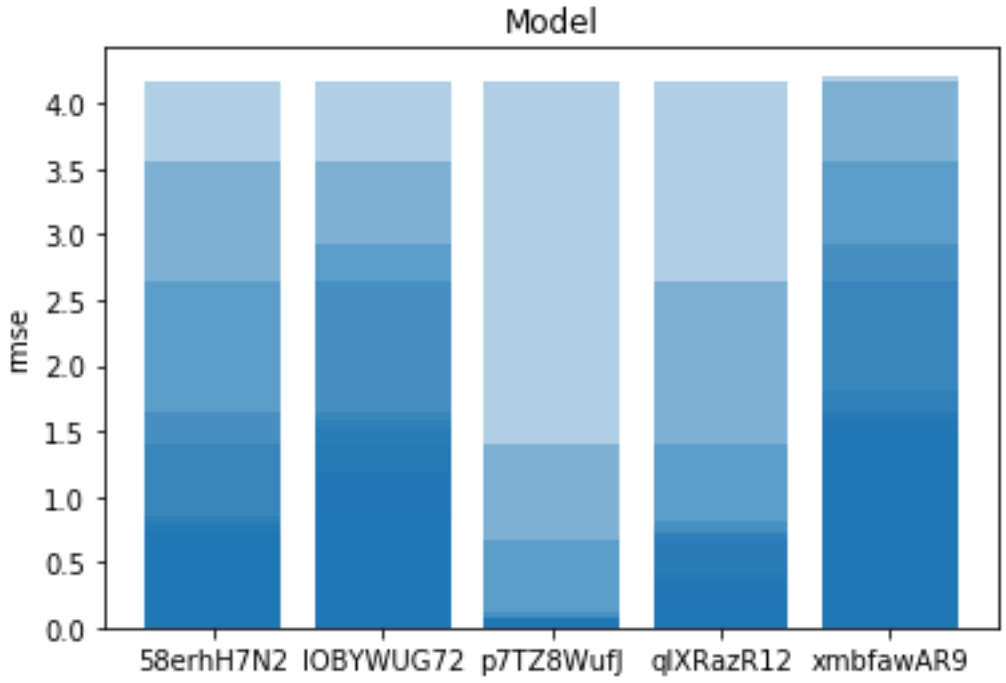
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

dataset_name	number_of runs	number_of_models	number_of_hyperparameters
CalCOFI - bottle.csv	5	5	10

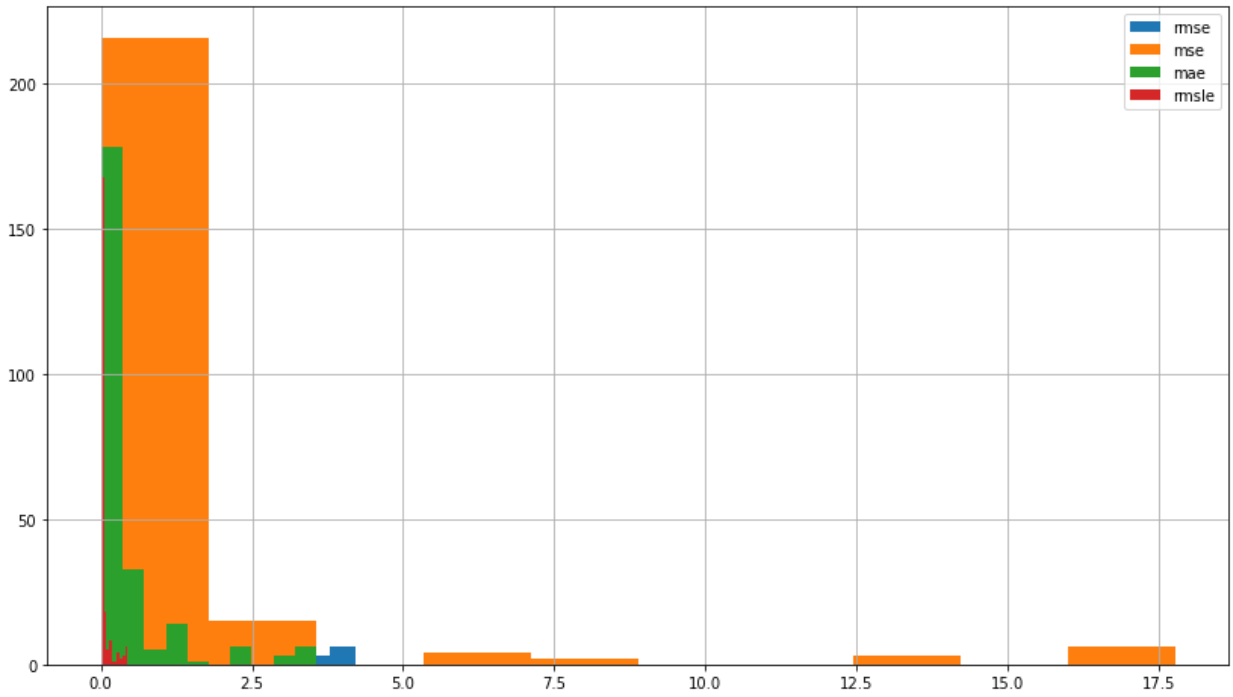
Darshan Durve
Shriram Karthikeyan

Analytics

Range of RMSE values for different run IDs

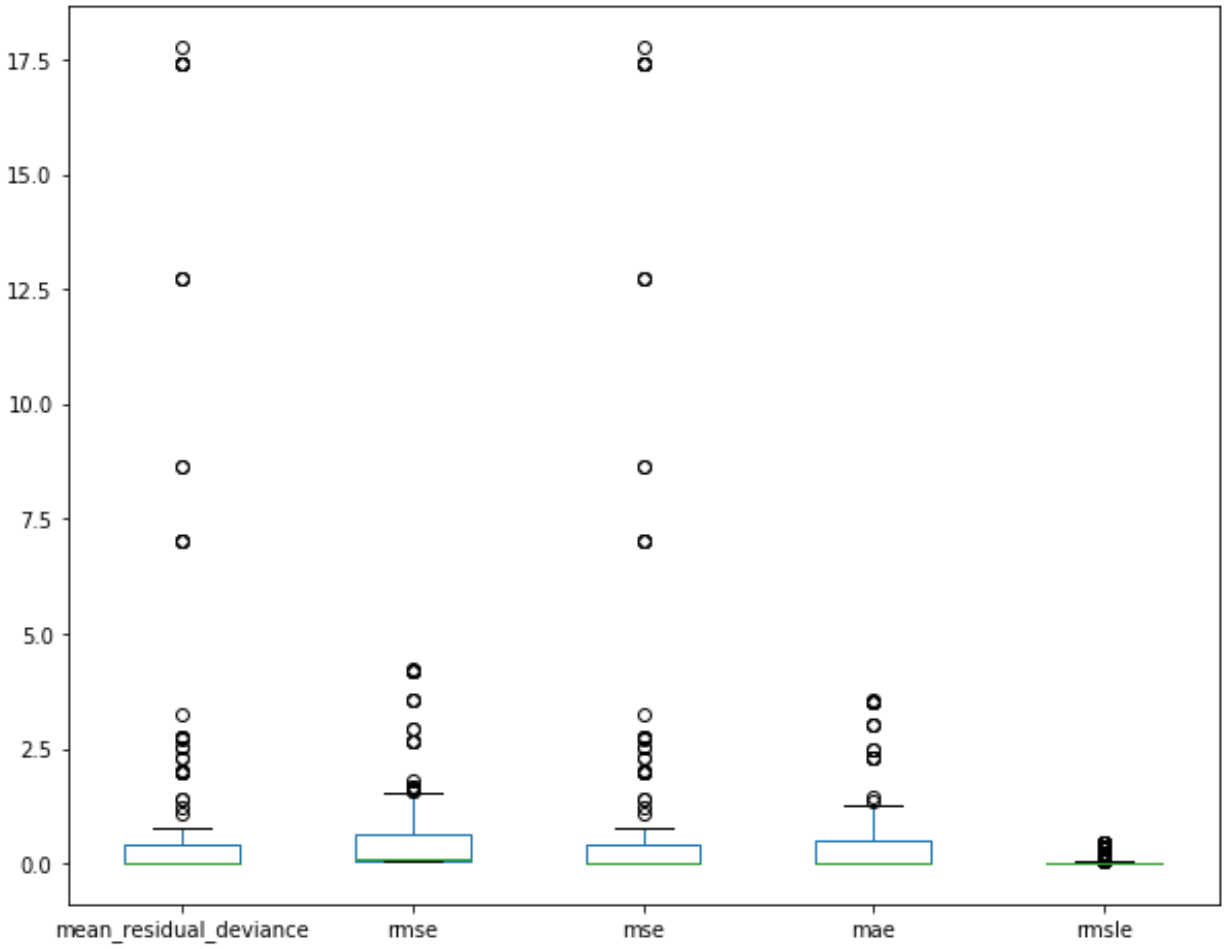


Frequency of Metrics



Darshan Durve
Shriram Karthikeyan

Box plot for performance metrics



Darshan Durve
Shriram Karthikeyan

Conclusion

This hyperparameter project has taught us how to model a database around a relatively new concept for us as database students. We have tried to create a database that can be implemented in a production environment to hold data about hyperparameters, models, and run for thousands of datasets and millions of records at the hyperparameter level. To make this project better we would like to delve deeper into making the functionality to search for tags and finding an appropriate dataset and suggestion of hyperparameters our focus.

Citations & References

https://stackoverflow.com/questions/19587118/iterating-through-directories-with-python
https://stackoverflow.com/questions/7099290/how-to-ignore-hidden-files-using-os-listdir
https://stackoverflow.com/questions/845058/how-to-get-line-count-cheaply-in-python
https://stackoverflow.com/questions/48996822/python-drop-rows-from-a-pandas-dataframe-that-contain-numbers
https://stackoverflow.com/questions/11346283/renaming-columns-in-pandashttps://www.nltk.org/book/ch05.html
https://www.nltk.org/book/ch05.html
https://stackoverflow.com/questions/2661778/tag-generation-from-a-text-content
https://stackoverflow.com/questions/14753321/add-auto-increment-id-to-existing-table
https://stackoverflow.com/questions/8384737/extract-file-name-from-path-no-matter-what-the-os-path-format
https://www.interviewqs.com/ddi_code_snippets/add_new_col_df_default_value

License