Hyperparameter project report

April 25, 2019

1 Hyperparameter project report

2 Abstract:

In hyperparameter database, our objective is to analyze the effect of hyperparameters on the following algorithms: Distributed random forest, generalized linear model, gradient boosting machine, naïve Bayes classifier and so on.

The hyperparameter database also uses these data to build models that can predict hyperparameters without search and for visualizing and teach statistical concepts such as power and bias/variance tradeoff.

3 Data source:

Our data source: https://www.kaggle.com/c/house-prices-advanced-regression-techniques, our training dataset has 81 columns, with 1460 records and our text dataset has 80 columns and with 1459 records.

4 Conceptual model:

our conceptual model has seven tables and we can clearly find the relationship between each table.

5 E-R diagram:

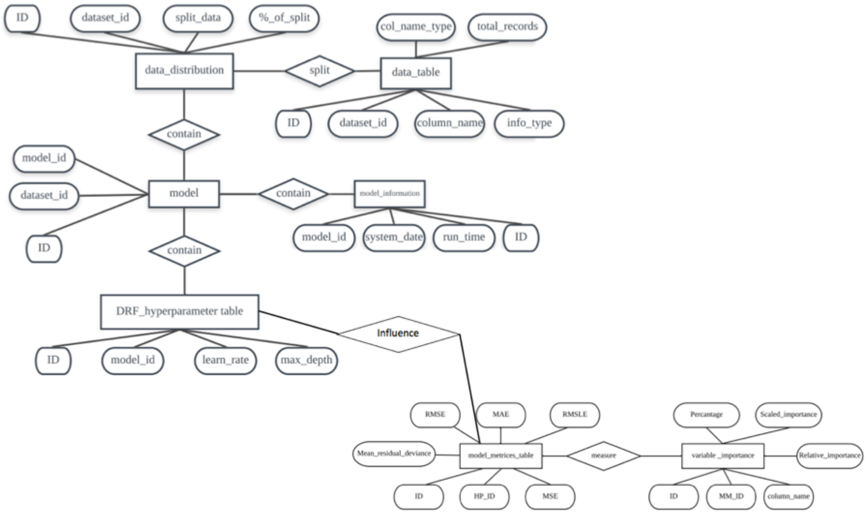
this our E-R diagram before we create the physical database.

6 Phycial model

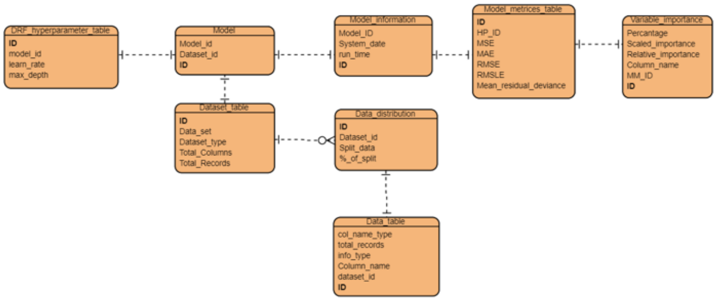
this the physical model after we upgrade our E-R diagram. we can see that we create 11 tables

7 Normalization

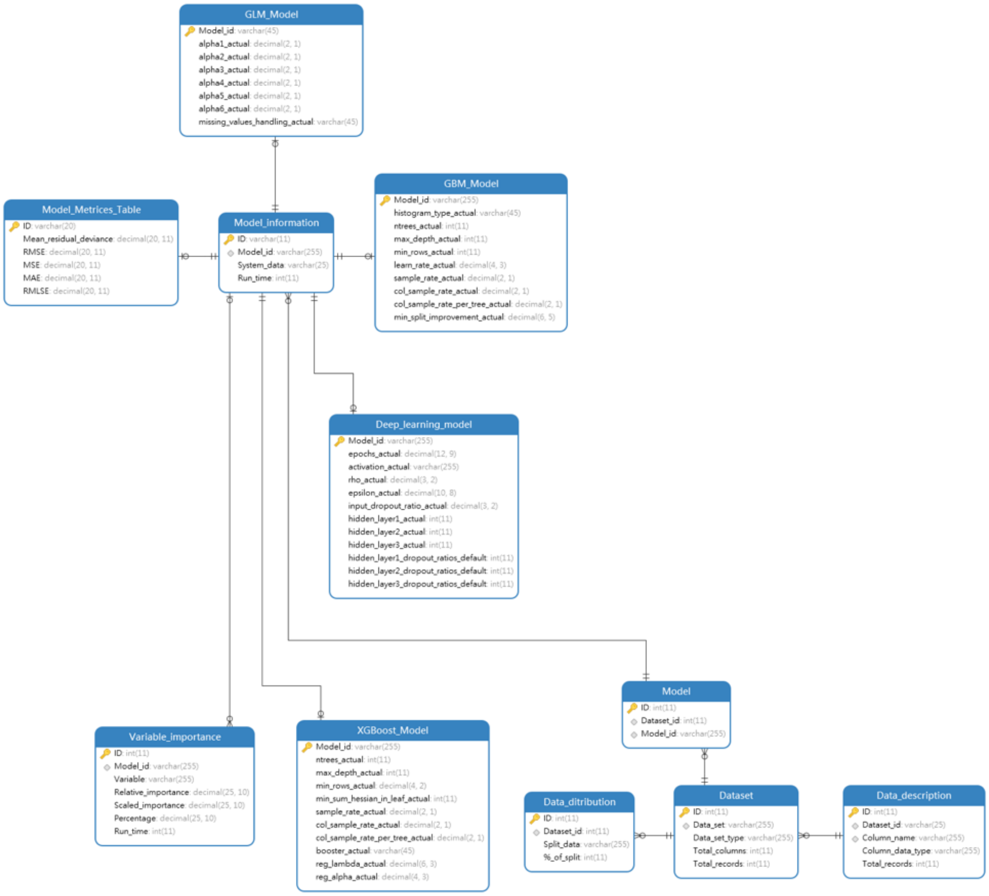
As the E-R diagram above illustrate, we creat 11 tables.In order to obey the 1st NF, we eliminate the repeated rows and when we get the dataset we find that some columns are not atomic


ID
dataset_id
split_data
%_of_split
col_name_type
total_records
data_distribution
split
data_table
ID
dataset_id
column_name
info_type
contain
model_id
dataset_id
model
contain
model_information
ID
model_id
system_date
run_time
ID
contain
DRF_hyperparameter table
Influence
ID
model_id
learn_rate
max_depth
RMSE
MAE
RMSLE
Percantage
Scaled_importance
Mean_residual_deviance
model_metrices_table
measure
variable _importance
Relative_importance
ID
HP_ID
MSE
ID
MM_ID
column_name

cm


DRF_hyperparameter_table
ID
model_id
learn_rate
max_depth
Model
Model_id
Dataset_id
ID
Model_information
Model_ID
System_date
run_time
ID
Model_metrices_table
ID
HP_ID
MSE
MAE
RMSE
RMSLE
Mean_residual_deviance
Variable_importance
Percantage
Scaled_importance
Relative_importance
Column_name
MM_ID
ID
Dataset_table
ID
Data_set
Dataset_type
Total_Columns
Total_Records
Data_distribution
ID
Dataset_id
Split_data
%_of_split
Data_table
col_name_type
total_records
info_type
Column_name
dataset_id
ID

er


GLM_Model
Model_id: varchar(45)
alpha1_actual: decimal(2, 1)
alpha2_actual: decimal(2, 1)
alpha3_actual: decimal(2, 1)
alpha4_actual: decimal(2, 1)
alpha5_actual: decimal(2, 1)
alpha6_actual: decimal(2, 1)
missing_values_handling_actual: varchar(45)
Model_Metrices_Table
ID: varchar(20)
Mean_residual_deviance: decimal(20, 11)
RMSE: decimal(20, 11)
MSE: decimal(20, 11)
MAE: decimal(20, 11)
RMLSE: decimal(20, 11)
Model_information
ID: varchar(11)
Model_id: varchar(255)
System_data: varchar(25)
Run_time: int(11)
GBM_Model
Model_id: varchar(255)
histogram_type_actual: varchar(45)
ntrees_actual: int(11)
max_depth_actual: int(11)
min_rows_actual: int(11)
learn_rate_actual: decimal(4, 3)
sample_rate_actual: decimal(2, 1)
col_sample_rate_actual: decimal(2, 1)
col_sample_rate_per_tree_actual: decimal(2, 1)
min_split_improvement_actual: decimal(6, 5)
Deep_learning_model
Model_id: varchar(255)
epochs_actual: decimal(12, 9)
activation_actual: varchar(255)
rho_actual: decimal(3, 2)
epsilon_actual: decimal(10, 8)
input_dropout_ratio_actual: decimal(3, 2)
hidden_layer1_actual: int(11)
hidden_layer2_actual: int(11)
hidden_layer3_actual: int(11)
hidden_layer1_dropout_ratios_default: int(11)
hidden_layer2_dropout_ratios_default: int(11)
hidden_layer3_dropout_ratios_default: int(11)
Model
ID: int(11)
Dataset_id: int(11)
Model_id: varchar(255)
Variable_importance
ID: int(11)
Model_id: varchar(255)
Variable: varchar(255)
Relative_importance: decimal(25, 10)
Scaled_importance: decimal(25, 10)
Percentage: decimal(25, 10)
Run_time: int(11)
XGBoost_Model
Model_id: varchar(255)
ntrees_actual: int(11)
max_depth_actual: int(11)
min_rows_actual: decimal(4, 2)
min_sum_hessian_in_leaf_actual: int(11)
sample_rate_actual: decimal(2, 1)
col_sample_rate_actual: decimal(2, 1)
col_sample_rate_per_tree_actual: decimal(2, 1)
booster_actual: varchar(45)
reg_lambda_actual: decimal(6, 3)
reg_alpha_actual: decimal(4, 3)
Data_ditribution
ID: int(11)
Dataset_id: int(11)
Split_data: varchar(255)
%_of_split: int(11)
Dataset
ID: int(11)
Data_set: varchar(255)
Data_set_type: varchar(255)
Total_columns: int(11)
Total_records: int(11)
Data_description
ID: int(11)
Dataset_id: varchar(25)
Column_name: varchar(255)
Column_data_type: varchar(255)
Total_records: int(11)

pm

so we divide it different columns and the we set up the primary key and foreign key to confirm that our database comply with the 2NF and 3NF.

8 10 use cases

8.0.1 1.Find the best model name and corresponding RMSE value which evaluate the model performance.

```
SELECT distinct a.ID ,a.Model_id,b.RMSE
FROM Model_information a, Model_Metrices_Table b ,Variable_importance c
WHERE a.ID=c.Model_id and a.ID=b.ID;
```

8.0.2 2.Find top 10 deeplearning model to find their evaluation performance.

```
SELECT a.ID,a.Model_id,b.RMSE
FROM Model_information a, Model_Metrices_Table b
WHERE Model_id like 'Deep%' and a.ID= b.ID
ORDER BY RMSE
limit 10;
```

8.0.3 3.Find one type of models to calculate its average RMSE value.

```
SELECT AVG(b.RMSE) as 'average performance'
FROM Model_information a, Model_Metrices_Table b
WHERE Model_id like 'GBM%'and a.ID= b.ID;
```

8.0.4 4.Find one model's one parameters.

```
SELECT a.Model_id,b.max_depth_actual
FROM Model_information a, XGBoost_Model b
WHERE a.ID= b.Model_id
ORDER BY max_depth_actual;
```

8.0.5 5.Find one model's RMSE value and its hyperparameters.

```
SELECT mm.ID, mm.Model_id, mm.Run_time,
me.RMSE,
dl. activation_actual, dl.rho_actual,epochs_actual
FROM Model_information mm
left join Deep_learning_model dl
ON mm.ID = dl.Model_id
JOIN Model_Metrices_Table me
ON  dl.Model_id = me.ID
WHERE mm.ID LIKE "DL%"
ORDER BY  mm.Run_time;
```

8.0.6 6.Find the model and its RMSE value which is more than one model's average RMSE value.

```
SELECT a.Model_id,b.RMSE
FROM Model_information a, Model_Metrices_Table b
WHERE a.ID=b.ID and b.RMSE<(
SELECT AVG(b.RMSE)
FROM Model_information a, Model_Metrices_Table b
WHERE Model_id like 'G%'and a.ID= b.ID)
ORDER BY b.RMSE
LIMIT 10;
```

8.0.7 7.When the run time is 1000, find the most important predictors which percentage is more than 0.2.

```
SELECT b.Variable
FROM Variable_importance b
where b.Percentage>0.2 and b.Model_id=(
SELECT distinct c.ID
FROM Model_information c,Variable_importance d
WHERE c.ID=d.Model_id and c.Run_time=1000);
```

8.0.8 8.Find the best model's top five important variables and their percentage.

```
select Model_id, Variable,Percentage
from Variable_importance
where (
   select count(*) from Variable_importance as f
   where f.Model_id = Variable_importance.Model_id
  AND f.Percentage > Variable_importance.Percentage)<=4
  ORDER BY Model_id;
```

8.0.9 9.How many models has run for 1000.

```
SELECT COUNT(*) as " the number of 1500 runtime model"
FROM Model_information mm
WHERE mm.Run_time = 1500
order BY mm.Run_time;
```

8.0.10 10.Calculate two model's average RMSE to find which model is better.

```
SELECT DISTINCT (
SELECT AVG(a.RMSE) FROM Model_Metrices_Table a WHERE a.ID like "DL%"
) - (
SELECT AVG(a.RMSE) FROM Model_Metrices_Table a WHERE a.ID like "XBG%"
) AS difference_between_two_model
FROM Model_Metrices_Table;
```

9 4 VIEWS:

9.0.1 1.Get top model performance

```
CREATE VIEW TOP_MODEL AS
SELECT ID,RMSE
FROM Model_Metrices_Table
ORDER BY RMSE DESC
LIMIT 5;
```

9.0.2 2.Get model information on run time and performance

```
CREATE VIEW Model_RMSE AS
SELECT Model_id,RMSE,Run_time
FROM Model_information a,Model_Metrices_Table b
WHERE a.ID=b.ID;
```

9.0.3 3.Get everydl model hyper in the db

```
CREATE VIEW DL_Hyper AS
SELECT a.Model_id,epochs_actual,RMSE
FROM Model_information a,Model_Metrices_Table b,Deep_learning_model c
WHERE a.ID=b.ID AND b.ID=c.Model_id AND c.Model_id=a.ID;
```

9.0.4 4.Get run time and variable importance of model.

```
CREATE VIEW Model_Variable_importance AS
SELECT a.Model_id,variable,Percentage,b.Run_time
FROM Model_information a,Variable_importance b
WHERE a.ID=b.Model_id;
```

10 4 Functions

10.0.1 1.Get moedl id

```
DELIMITER $$
CREATE FUNCTION getmodelid ( v_id VARCHAR(30)) RETURNS VARCHAR ( 255 ) BEGIN
    DECLARE
        modelid VARCHAR ( 255 );
    SELECT
        model_id INTO modelid
    FROM
        model_information
    WHERE
        ID = v_id;
    RETURN modelid;

END $$
```

10.0.2 2.Get performance of deep learning

```
DELIMITER $$
CREATE FUNCTION dlperformance() RETURNS VARCHAR(25) BEGIN
  DECLARE
        a VARCHAR(25);
    SELECT
        AVG(RMSE) INTO a
    FROM
        model_metrices_table
    WHERE
        ID LIKE 'DL%';
    RETURN a;

END $$
```

11 3.Get number of model smaller than RMSE

```
DELIMITER $$
CREATE FUNCTION NUMRMSE(num DECIMAL(5)) RETURNS INT BEGIN
  DECLARE
        a INT;
    SELECT
        COUNT(*) INTO a
    FROM
        model_metrices_table
    WHERE
        RMSE < num;
    RETURN a;

END $$
```

12 4.Get number of model in runtime

```
DELIMITER $$
CREATE FUNCTION runtimemodel(num INT) RETURNS INT BEGIN
  DECLARE
        a INT;
    SELECT
        COUNT(*) INTO a
    FROM
        model_information
    WHERE
        Run_time = num;
    RETURN a;

END $$
```

13 MIT License

```
In [ ]:
```