

HYPERPARAMETER SEARCH

Hyperparameter-db-project-db15

Pallavi Kashyap | INFO6210 | April 24, 2019

ABSTRACT

- Hyperparameter tuning is choosing a set of optimal hyperparameters for a learning algorithm. Hyperparameter tuning is critical for building accurate models, yet most researchers use personal experience to specify their range.

There are two different methods for optimizing hyperparameters:

Grid Search

Random Search

INTRODUCTION

This project is to build the database from thousands of public classification and regression dataset using the Distributed Random Forest (DRF), Generalized Linear Model (GLM), Gradient Boosting Machine (GBM) and XGBOOST algorithms. This data will allow us to estimate the relative importance of hyperparameters, appropriate hyperparameter ranges and to build predictive models of hyperparameter values. Out of those thousands, I am only working on one Dataset-Adult.csv from KAGGLE. After finding the Hyperparameters, we are storing the data in the Database for further analysis.

DESCRIPTION OF THIS DATASET

The link to the dataset: <https://www.kaggle.com/wenruliu/adult-income-dataset/version/2?#adult.csv>

This data was extracted from the census bureau database found by Donor: Ronny Kohavi and Barry Becker. This is a widely cited KNN dataset. I encountered it during my course, and I wish to share it here because it is a good starter example for data pre-processing and machine learning practices.

Fields The dataset contains 16 columns Target filed: Income -- The income is divided into two classes: $\leq 50K$ and $> 50K$

Number of attributes: 14 -- These are the demographics and other features to describe a person

We can explore the possibility in predicting income level based on the individual's personal information. A set of reasonably clean records was extracted using the following conditions: ((AAGE>16) && (AGI>100) && (AFNLWGT>1) && (HRSWK>0))
Prediction task is to determine whether a person makes over 50K a year.

ATTRIBUTE INFORMATION:

48842 instances mix of continuous and discrete (train=32561, test=16281)
5222 if instances with unknown values are removed (train=30162, test=15060)

Listing of attributes:

>50K, <=50K.

age: continuous.

workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.

fnlwgt: continuous.

education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.

education-num: continuous.

marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspect, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.

sex: Female, Male.

capital-gain: continuous.

capital-loss: continuous.

hours-per-week: continuous.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc.), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinidad & Tobago, Peru, Hong, Holland-Netherlands.

Duplicate or conflicting instances: 6

Probability for the label '>50K': 23.93% / 24.78% (without unknowns)

Probability for the label '<=50K': 76.07% / 75.22% (without unknowns)

TOOLS & APPLICATION USED

H2O: An open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform to build machine learning models and provide easy productionalization of those models.

Microsoft SQL Server: A relational database management system software. It is used for storing the Hyperparameters data.

DATA SCIENCE PART

We are using H2O which is an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform to build machine learning models and provide easy productionalization of those models.

To search for Hyperparameter, the AUTOML run on the whole dataset, using different run-ids. The result gives us best model from the Leaderboard which has performed better in-terms of AUC, RMSE, MSE, LOGLOSS metrics. Further, the model's performance is analyzed using Grid Search-where we hard coded the values of the hyperparameter for the best machine learning model and checked the AUC. For Example, if the best model in the leaderboard is 'GBM' then the Grid Search for GBM is done using H2OGradient Estimator.

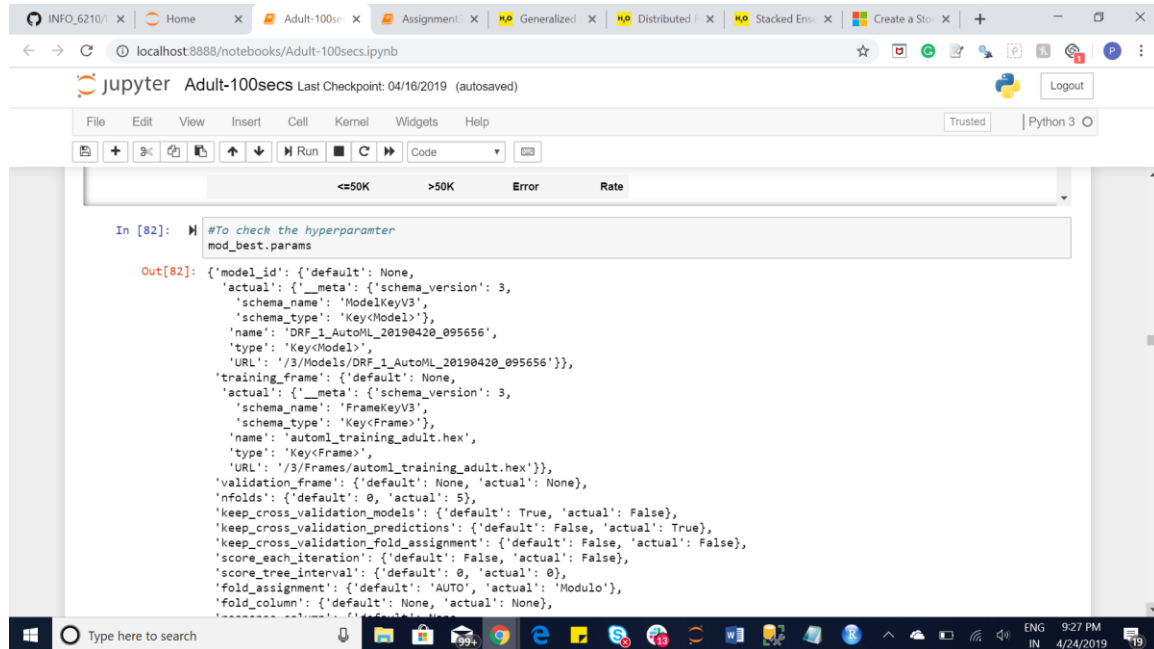
To find the range of the Hyperparameters, the Grid Search for GBM is done only for one hyperparameter by keeping others constant. This is done for all the Hyperparameters. There are other models in the leaderboard such as XRT, DRF, GLM, Deep Learning for which similar Grid Search is run to find the values and range of the Hyperparameters of these respective models.

This process is repeated for three to four different run-times using different datasets. After getting the required values and range of the Hyperparameters, the data is saved in Data frames and then is ready to store in the Database.

Automl run



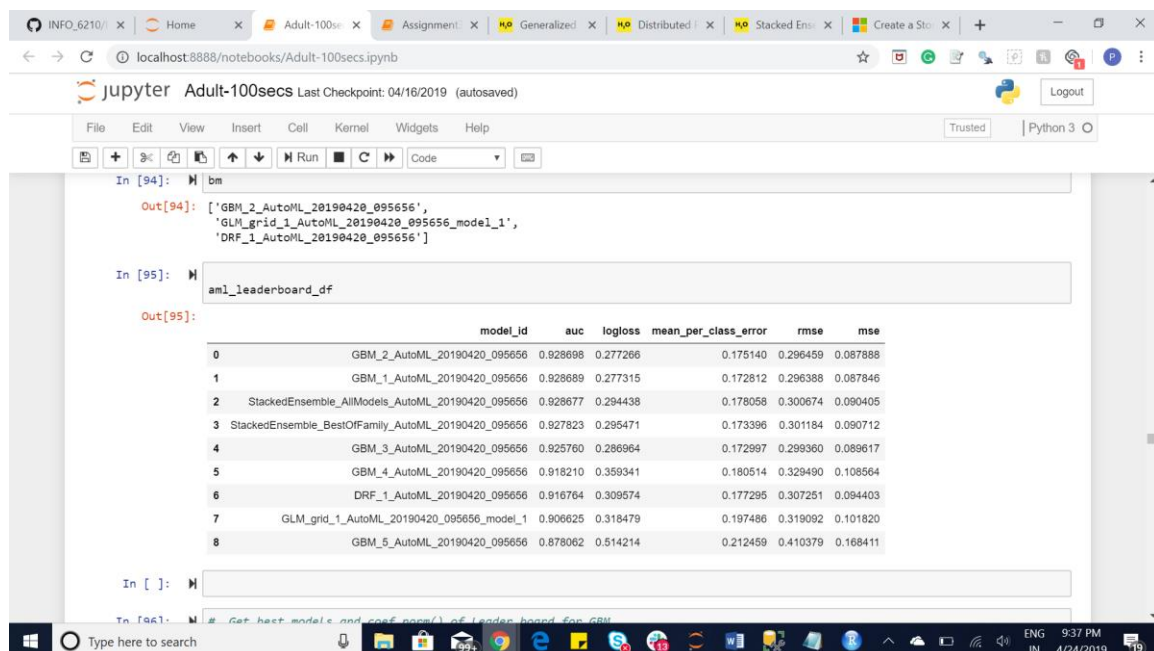
Json file containing Hyperparameters for the best model



```
In [82]: #To check the hyperparameter
mod_best.params

Out[82]: {'model_id': {'default': None,
  'actual': {'__meta': {'schema_version': 3,
    'schema_name': 'ModelKeyV3',
    'schema_type': 'Key<Model>'},
    'name': 'DRF_1_AutoML_20190420_095656',
    'type': 'Key<Model>',
    'URL': '/3/Models/DRF_1_AutoML_20190420_095656'}},
  'training_frame': {'default': None,
    'actual': {'__meta': {'schema_version': 3,
      'schema_name': 'FrameKeyV3',
      'schema_type': 'Key<Frame>'},
      'name': 'automl_training_adult.hex',
      'type': 'Key<Frame>',
      'URL': '/3/Frames/automl_training_adult.hex'}},
    'validation_frame': {'default': None, 'actual': None},
    'n_folds': {'default': 0, 'actual': 5},
    'keep_cross_validation_models': {'default': True, 'actual': False},
    'keep_cross_validation_predictions': {'default': False, 'actual': True},
    'keep_cross_validation_fold_assignment': {'default': False, 'actual': False},
    'score_each_iteration': {'default': False, 'actual': False},
    'score_tree_interval': {'default': 0, 'actual': 0},
    'fold_assignment': {'default': 'AUTO', 'actual': 'Modulo'},
    'fold_column': {'default': None, 'actual': None},
    'fold_per_column': {'default': None, 'actual': None}}
```

Leaderboard



```
In [94]: bm

Out[94]: ['GBM_2_AutoML_20190420_095656',
  'GLM_grid_1_AutoML_20190420_095656_model_1',
  'DRF_1_AutoML_20190420_095656']

In [95]: aml_leaderboard_df

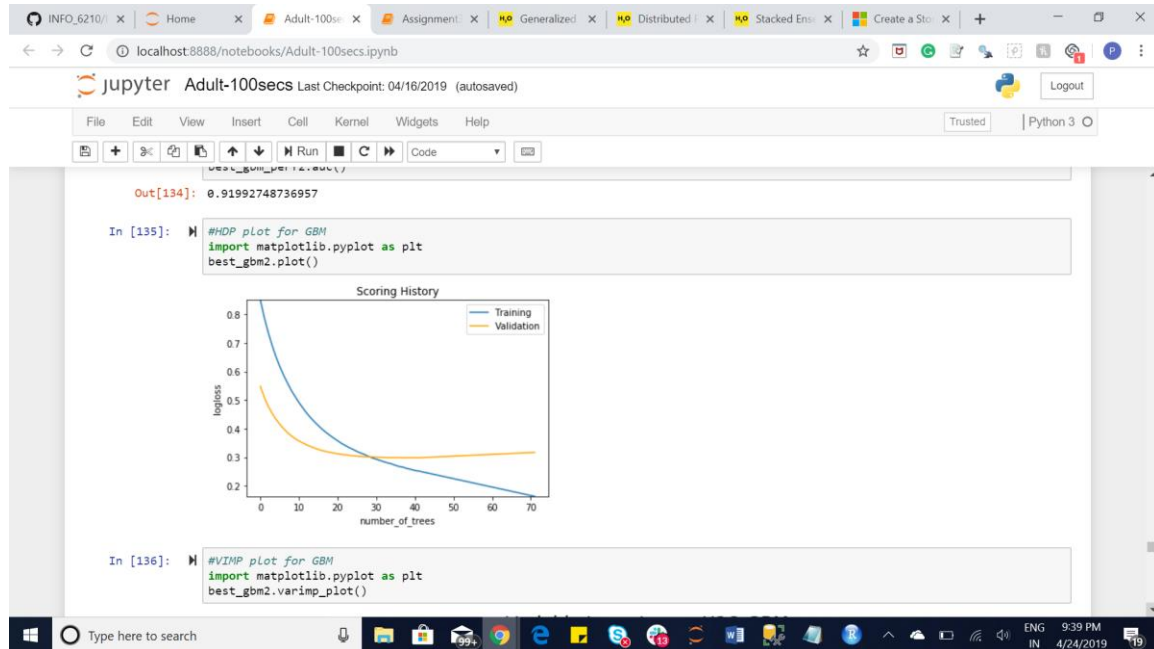
Out[95]:
```

	model_id	auc	logloss	mean_per_class_error	rmse	mse
0	GBM_2_AutoML_20190420_095656	0.928698	0.277266	0.175140	0.296459	0.087888
1	GBM_1_AutoML_20190420_095656	0.928689	0.277315	0.172812	0.296388	0.087846
2	StackedEnsemble_AllModels_AutoML_20190420_095656	0.928677	0.294438	0.178058	0.300674	0.090405
3	StackedEnsemble_BestOfFamily_AutoML_20190420_095656	0.927823	0.295471	0.173396	0.301184	0.090712
4	GBM_3_AutoML_20190420_095656	0.925760	0.286964	0.172997	0.299360	0.089617
5	GBM_4_AutoML_20190420_095656	0.918210	0.359341	0.180514	0.329490	0.108564
6	DRF_1_AutoML_20190420_095656	0.916764	0.309574	0.177295	0.307251	0.094403
7	GLM_grid_1_AutoML_20190420_095656_model_1	0.906625	0.319479	0.197486	0.319092	0.101620
8	GBM_5_AutoML_20190420_095656	0.878062	0.514214	0.212459	0.410379	0.168411

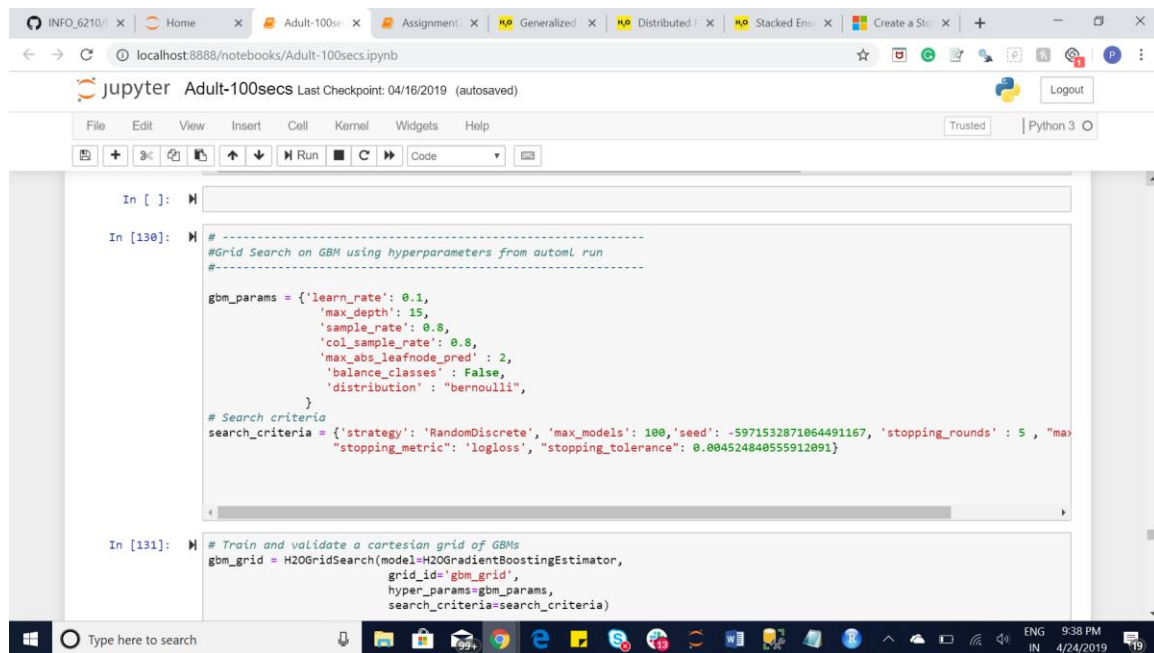
```
In [ ]:
```

```
In [96]: # Get best model and conf point of leader board for GBM
```

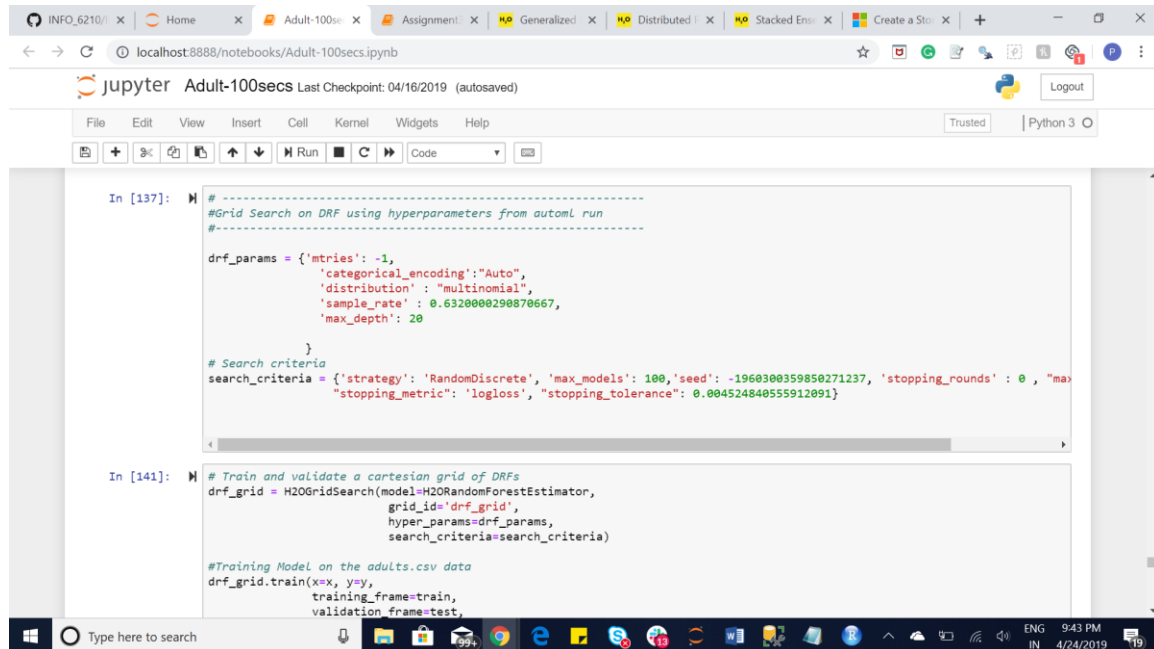
Grid Search for GBM



Grid Search for GBM



Grid Search for DRF



The screenshot shows a Jupyter Notebook window titled "Adult-100secs" with a last checkpoint of "04/16/2019 (autosaved)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running cells, and code execution. The code is written in Python and is divided into two cells. The first cell, labeled "In [137]:", contains a comment "# Grid Search on DRF using hyperparameters from automl run" and defines a dictionary of hyperparameters for a Decision Random Forest (DRF). The second cell, labeled "In [141]:", contains a comment "# Train and validate a cartesian grid of DRFs" and defines an H2OGridSearch object with the hyperparameters from the first cell. The code is as follows:

```
In [137]: # -----
# Grid Search on DRF using hyperparameters from automl run
# -----

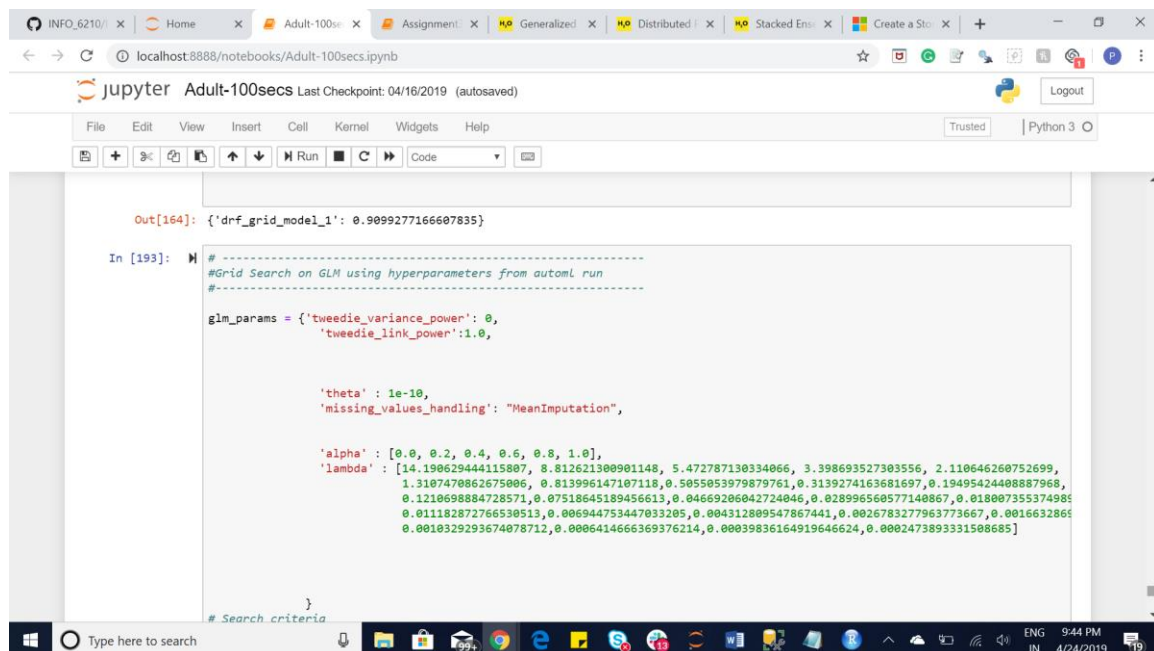
drf_params = {'mtries': -1,
              'categorical_encoding': "Auto",
              'distribution': "multinomial",
              'sample_rate': 0.6320000290870667,
              'max_depth': 20
            }

# Search criteria
search_criteria = {'strategy': 'RandomDiscrete', 'max_models': 100, 'seed': -1960300359850271237, 'stopping_rounds': 0, "max_stopping_metric": 'logloss', "stopping_tolerance": 0.004524840555912091}

In [141]: # Train and validate a cartesian grid of DRFs
drf_grid = H2OGridSearch(model=H2ORandomForestEstimator,
                        grid_id='drf_grid',
                        hyper_params=drf_params,
                        search_criteria=search_criteria)

# Training Model on the adults.csv data
drf_grid.train(x=x, y=y,
              training_frame=train,
              validation_frame=test,
```

Grid Search for GLM



The screenshot shows a Jupyter Notebook window titled "Adult-100secs" with a last checkpoint of "04/16/2019 (autosaved)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running cells, and code execution. The code is written in Python and is divided into two cells. The first cell, labeled "Out[164]:", shows the output of a previous cell, which is a dictionary containing the best model ID. The second cell, labeled "In [193]:", contains a comment "# Grid Search on GLM using hyperparameters from automl run" and defines a dictionary of hyperparameters for a Generalized Linear Model (GLM). The code is as follows:

```
Out[164]: {'drf_grid_model_1': 0.9099277166607835}

In [193]: # -----
# Grid Search on GLM using hyperparameters from automl run
# -----

glm_params = {'tweedie_variance_power': 0,
              'tweedie_link_power': 1.0,

              'theta': 1e-10,
              'missing_values_handling': "MeanImputation",

              'alpha': [0.0, 0.2, 0.4, 0.6, 0.8, 1.0],
              'lambda': [14.190629444115807, 8.812621300901148, 5.472787130334066, 3.398693527303556, 2.110646260752699,
                        1.3107470862675006, 0.813996147107118, 0.5055053979879761, 0.3139274163681697, 0.19495424408887968,
                        0.1210698884728571, 0.07518645189456613, 0.04669206042724046, 0.028996560577140867, 0.018007355374985,
                        0.011182872766530513, 0.006944753447033205, 0.004312809547867441, 0.0026783277963773667, 0.0016632865,
                        0.0010329293674078712, 0.0006414666369376214, 0.00039836164919646624, 0.0002473893331508685]
            }

# Search criteria
```

DATABASE STORAGE PART

After find the Hyperparameters from all the run times using different combinations, the data is merged for respective models in data frames. These data frames then saved as csv to be imported in SQL Server.

The conceptual diagram and E-R diagram is created which provides information about the relationship, fields and attribute for the table.

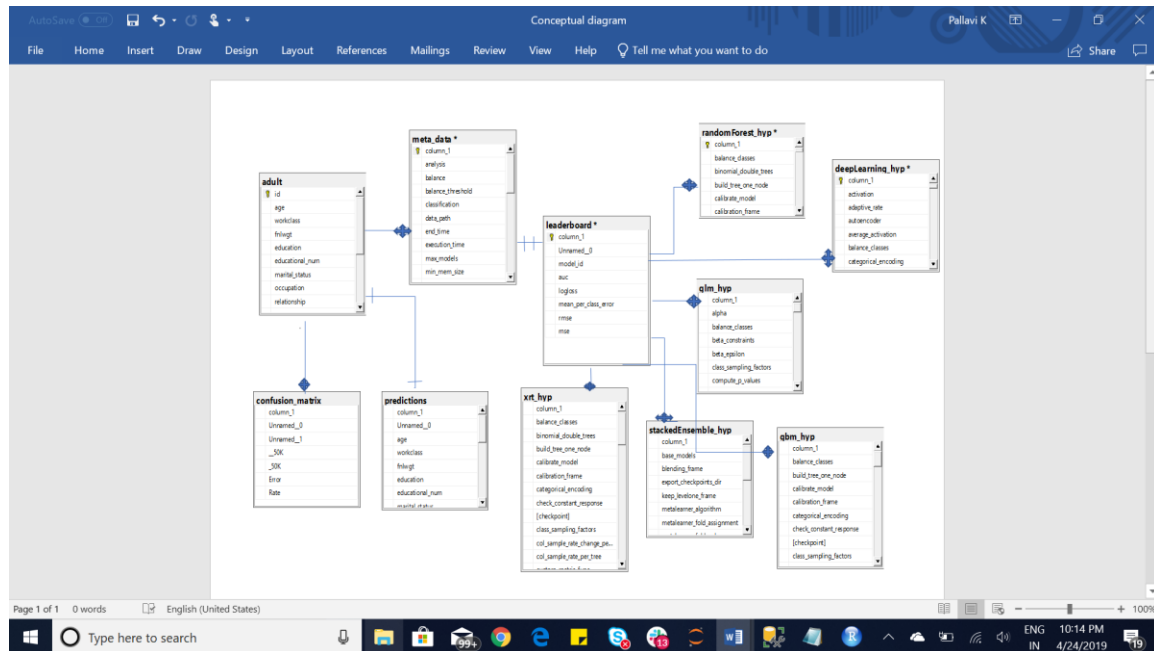
The tables are created according to the conceptual diagram for leaderboard, metadata, prediction and respective models etc. After the tables are created then according to project requirement, 5 Use Cases, 5 Views, 5 Function 5 procedure and 5 Index.

The five Use Cases are :

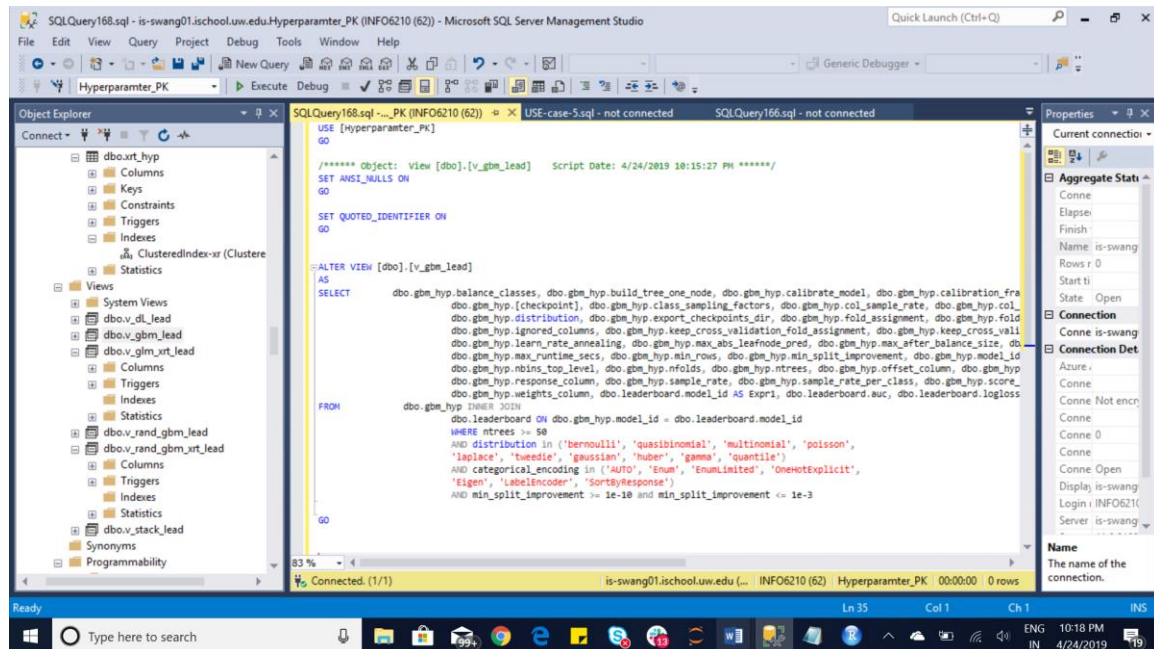
1. Compare the metrics and also find the common Hyperparamter for like models such GBM, RANDOM FOREST, XRT
2. What are the metrics for stacked ensemble model where base models are present for every model id
3. Find the Hyperparameters from GBM where number of trees are greater than 50 and distribution is either Bernoulli or multinomial
min_split_improvement is between $1e-10$ and $1e-3$ and
4. Selecting best DRF and GLM , Deeplearning, XRT, Stacked Ensemble, GBM model
5. Range of Hyperparamter and othe parameters for GBM

Below are the screen shot for the Database part:

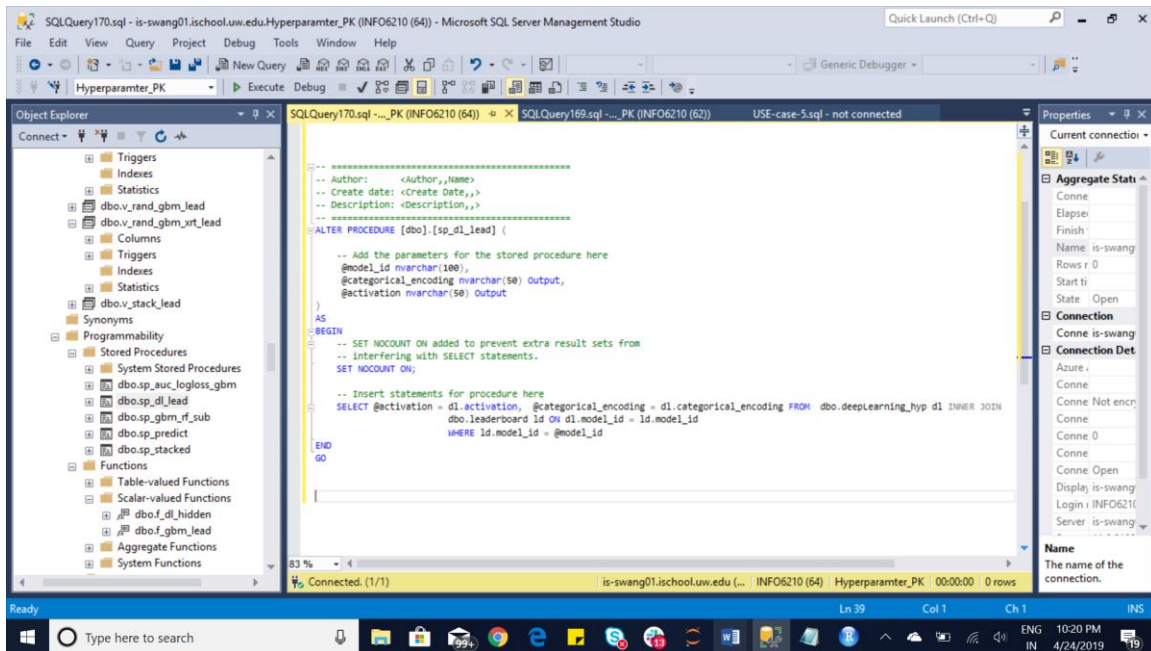
Conceptual Diagram



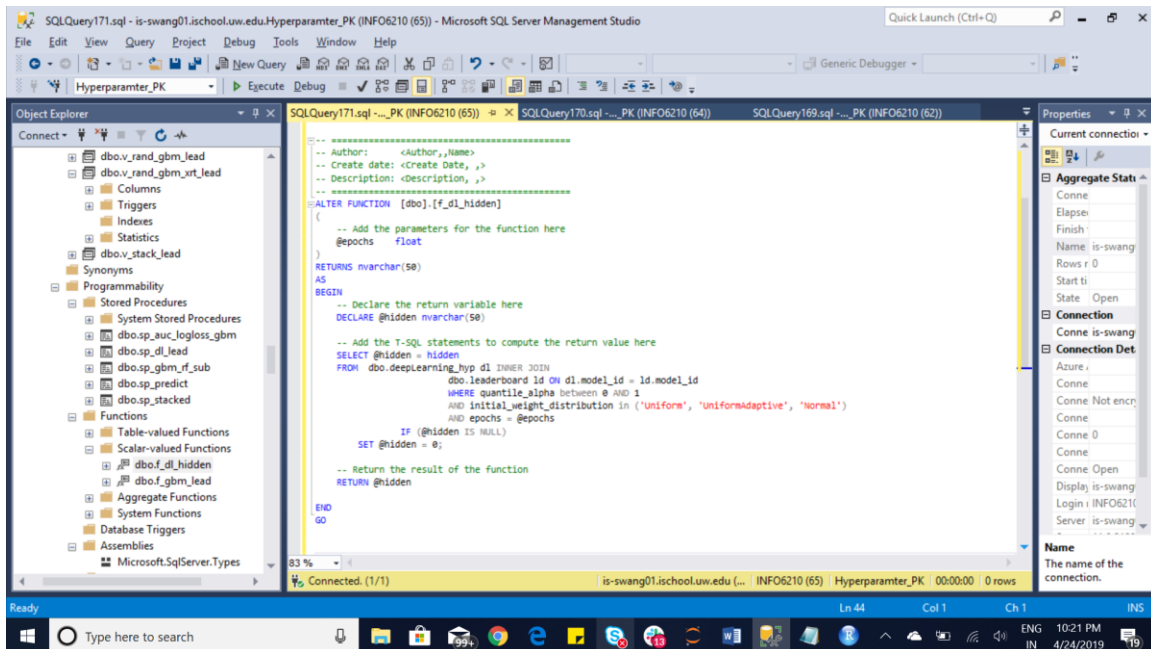
View



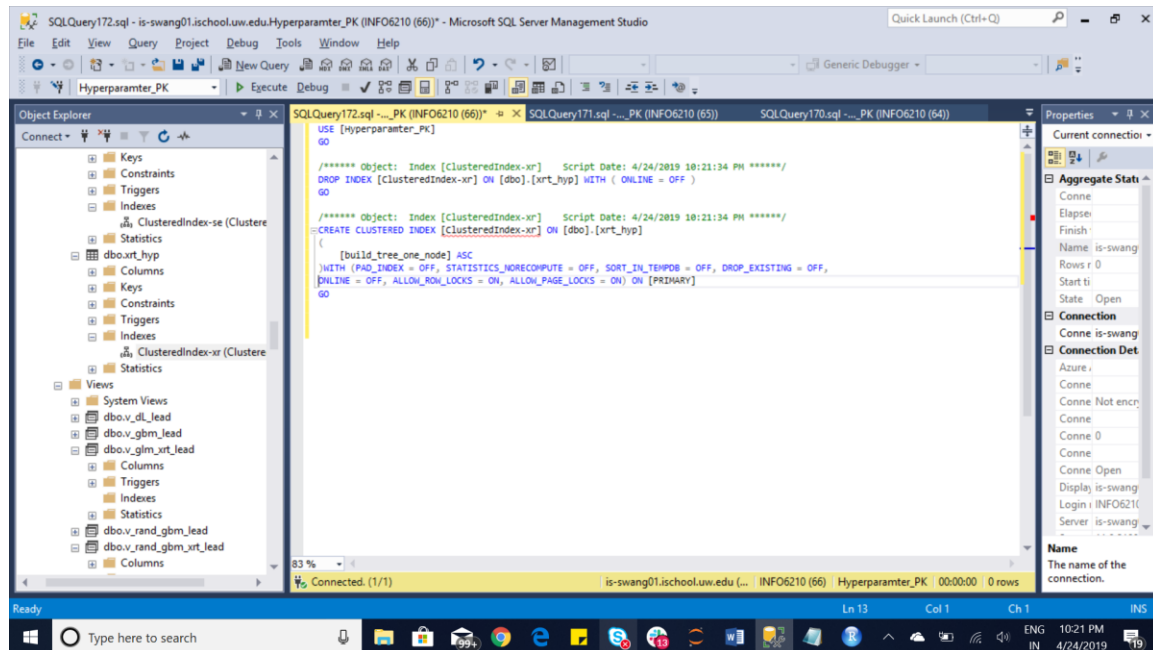
Procedure:



Functions:



Index:



CONCLUSION

These hyperparameter data will allow to estimate the relative importance of hyperparameter values. These values will be used as a backend database of a Website which will allow users to search, visualize, and explore the data.

REFERENCES

<https://towardsdatascience.com/hyperparameter-tuning-c5619e7e6624>
<https://www.kaggle.com/wenruihu/adult-income-dataset/version/2?#adult.csv>
https://github.com/prabhushub/HyperparameterSamples/blob/master/Meta_Data_Generated/StudentPerformanceDataset/StudentPerformance_100.ipynb
https://github.com/nikbearbrown/CSYE_7245/blob/master/H2O/H2O_automl_model.ipynb
<http://docs.h2o.ai/h2o/latest-stable/h2o-docs/grid-search.html>
<https://opensource.org/licenses/MIT>