Problem 1 - Hyperparameter Optimization using H20 20 points

In this question, you will compare the performances of H2O's grid search and randomized grid search. You will use the H2ORandomForestEstimator model, and use the allyears2k headers.zip dataset used in the this link in the classification example.

1. Grid search

(a) Perform grid search for identifying the best hyperparameters for the H2ORandomForestEstimator model with 'ntrees':[10,30,50,100] and 'max depth': [1,2,4,6]. (2)

(b) Display the grid results, sorted by accuracy in a decreasing order. (2)

(c) Identify the best model and evaluate the model's performance on a test set and display the AUC score.

In [ ]:
```
! pip install h2o
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-w
heels/public/simple/
Requirement already satisfied: h2o in /usr/local/lib/python3.8/dist-packages
(3.38.0.3)
Requirement already satisfied: future in /usr/local/lib/python3.8/dist-package
s (from h2o) (0.16.0)
Requirement already satisfied: requests in /usr/local/lib/python3.8/dist-packa
ges (from h2o) (2.23.0)
Requirement already satisfied: tabulate in /usr/local/lib/python3.8/dist-packa
ges (from h2o) (0.8.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/
dist-packages (from requests->h2o) (2022.9.24)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /us
r/local/lib/python3.8/dist-packages (from requests->h2o) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.8/d
ist-packages (from requests->h2o) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-p
ackages (from requests->h2o) (2.10)
```

In [ ]:
```
import h2o
from h2o.automl import H2OAutoML
h2o.init(nthreads = -1, max_mem_size = 8)
h2o.connect()

airlines= h2o.import_file("https://s3.amazonaws.com/h2o-public-test-data/smal

# set the factors:
airlines["Year"]= airlines["Year"].asfactor()
airlines["Month"]= airlines["Month"].asfactor()
airlines["DayOfWeek"] = airlines["DayOfWeek"].asfactor()
airlines["Cancelled"] = airlines["Cancelled"].asfactor()
airlines['FlightNum'] = airlines['FlightNum'].asfactor()

# set the predictors and response columns:
predictors = ["Origin", "Dest", "Year", "UniqueCarrier",
              "DayOfWeek", "Month", "Distance", "FlightNum"]
response = "IsDepDelayed"
```

```
Checking whether there is an H2O instance running at http://localhost:54321
..... not found.
Attempting to start a local H2O server...
  Java Version: openjdk version "11.0.17" 2022-10-18; OpenJDK Runtime Environm
ent (build 11.0.17+8-post-Ubuntu-1ubuntu218.04); OpenJDK 64-Bit Server VM (bui
ld 11.0.17+8-post-Ubuntu-1ubuntu218.04, mixed mode, sharing)
  Starting server from /usr/local/lib/python3.8/dist-packages/h2o/backend/bin/
h2o.jar
  Ice root: /tmp/tmpcaj141z_
  JVM stdout: /tmp/tmpcaj141z_/h2o_unknownUser_started_from_python.out
  JVM stderr: /tmp/tmpcaj141z_/h2o_unknownUser_started_from_python.err
  Server is running at http://127.0.0.1:54321
Connecting to H2O server at http://127.0.0.1:54321 ... successful.
```

H2O_cluster_uptime:	04 secs
H2O_cluster_timezone:	Etc/UTC
H2O_data_parsing_timezone:	UTC
H2O_cluster_version:	3.38.0.3
H2O_cluster_version_age:	17 days
H2O_cluster_name:	H2O_from_python_unknownUser_9xh3av
H2O_cluster_total_nodes:	1
H2O_cluster_free_memory:	8 Gb
H2O_cluster_total_cores:	2
H2O_cluster_allowed_cores:	2
H2O_cluster_status:	locked, healthy
H2O_connection_url:	http://127.0.0.1:54321
H2O_connection_proxy:	{"http": null, "https": null}
H2O_internal_security:	False
Python_version:	3.8.16 final

```
Connecting to H2O server at http://localhost:54321 ... successful.
```

H2O_cluster_uptime:	05 secs
H2O_cluster_timezone:	Etc/UTC
H2O_data_parsing_timezone:	UTC
H2O_cluster_version:	3.38.0.3
H2O_cluster_version_age:	17 days
H2O_cluster_name:	H2O_from_python_unknownUser_9xh3av
H2O_cluster_total_nodes:	1
H2O_cluster_free_memory:	8 Gb
H2O_cluster_total_cores:	2
H2O_cluster_allowed_cores:	2
H2O_cluster_status:	locked, healthy
H2O_connection_url:	http://localhost:54321
H2O_connection_proxy:	{"http": null, "https": null}
H2O_internal_security:	False

Python_version: 3.8.16 final

```
Parse progress: |████████████████████████████████████████████████████████
████| (done) 100%
```

In [ ]:
```
train, valid = airlines.split_frame(ratios = [.8], seed = 1234)
```

In [ ]:
```
from h2o.estimators import H2ORandomForestEstimator
from h2o.grid.grid_search import H2OGridSearch

hyper_parameters = { 'ntrees':[10,30,50,100],
                     'max_depth': [1,2,4,6] }

model_grid_search = H2OGridSearch(H2ORandomForestEstimator( nfolds=5),
      hyper_parameters,
      grid_id="random_plus_manual")

model_grid_search.train(x=predictors,y=response, training_frame=train,validat
```

```
drf Grid Build progress: |████████████████████████████████████████████████
████| (done) 100%
```

Out[ ]:

Hyper-Parameter Search Summary: ordered by increasing logloss

max_depth	ntrees	model_ids	logloss
6.0	100.0	random_plus_manual_model_16	0.6074097
6.0	30.0	random_plus_manual_model_8	0.6080527
6.0	50.0	random_plus_manual_model_12	0.6084735
6.0	10.0	random_plus_manual_model_4	0.6113018
4.0	50.0	random_plus_manual_model_11	0.6246213
4.0	30.0	random_plus_manual_model_7	0.6250392
4.0	100.0	random_plus_manual_model_15	0.6253382
4.0	10.0	random_plus_manual_model_3	0.6254629
2.0	100.0	random_plus_manual_model_14	0.6497448
2.0	50.0	random_plus_manual_model_10	0.6497561
2.0	30.0	random_plus_manual_model_6	0.6503539
2.0	10.0	random_plus_manual_model_2	0.6514182
1.0	50.0	random_plus_manual_model_9	0.6668661
1.0	100.0	random_plus_manual_model_13	0.6676622
1.0	10.0	random_plus_manual_model_1	0.6687254
1.0	30.0	random_plus_manual_model_5	0.6696191

In [ ]:
```
Rand_forest_model = model_grid_search.get_grid(sort_by='auc', decreasing=True
Rand_forest_model
```

Out[ ]:

Hyper-Parameter Search Summary: ordered by decreasing auc

max_depth	ntrees	model_ids	auc
6.0	100.0	random_plus_manual_model_16	0.7356001
6.0	50.0	random_plus_manual_model_12	0.7344319

max_depth	ntrees	model_ids	auc
6.0	30.0	random_plus_manual_model_8	0.7330098
6.0	10.0	random_plus_manual_model_4	0.7290180
4.0	100.0	random_plus_manual_model_15	0.7199165
4.0	30.0	random_plus_manual_model_7	0.7178729
4.0	50.0	random_plus_manual_model_11	0.7172048
4.0	10.0	random_plus_manual_model_3	0.7165073
2.0	50.0	random_plus_manual_model_10	0.6919059
2.0	100.0	random_plus_manual_model_14	0.6904609
2.0	30.0	random_plus_manual_model_6	0.6888067
2.0	10.0	random_plus_manual_model_2	0.6750081
1.0	100.0	random_plus_manual_model_13	0.6639136
1.0	30.0	random_plus_manual_model_5	0.6636095
1.0	50.0	random_plus_manual_model_9	0.6601883
1.0	10.0	random_plus_manual_model_1	0.6531592

In [ ]:
```
best_model = Rand_forest_model.models[0]

# Now let's evaluate the model performance on a test set
# so we get an honest estimate of top model performance
best_model_perf1 = best_model.model_performance(valid)

best_model_perf1.auc()
```

Out[ ]:
```
0.7341018483113351
```

1. Randomized grid search

(a) Using the same model and hyperparameters grid, perform hyperparameter optimization using randomized grid search. Use a maximum of 10 models. (2)

(b) Display the results sorted by accuracy in a decreasing order. (2)

(c) Identify the best model and evaluate the model's performance on a test set and display the auc score. (2)

In [ ]:
```
search_criteria = { 'strategy': "RandomDiscrete", 'seed': 42,
                    'stopping_tolerance': 0.001,
                    'max_models': 10,
                   'max_runtime_secs': 120}
model_rand_search = H2OGridSearch(H2ORandomForestEstimator( nfolds=5),
      hyper_parameters,
      grid_id="random_plus_manual",
      search_criteria=search_criteria)

model_rand_search.train(x=predictors,y=response, training_frame=train)
```

```
drf Grid Build progress: |
| (done) 100%
```

Out[ ]:

Hyper-Parameter Search Summary: ordered by increasing logloss

max_depth	ntrees	model_ids	logloss
6.0	100.0	random_plus_manual_model_16	0.6074097
6.0	30.0	random_plus_manual_model_8	0.6080527
6.0	100.0	random_plus_manual_model_22	0.6082100
6.0	50.0	random_plus_manual_model_12	0.6084735
6.0	30.0	random_plus_manual_model_20	0.6085294
6.0	50.0	random_plus_manual_model_28	0.6087198
6.0	10.0	random_plus_manual_model_4	0.6113018
6.0	10.0	random_plus_manual_model_23	0.6114164
4.0	30.0	random_plus_manual_model_26	0.6244080
4.0	50.0	random_plus_manual_model_11	0.6246213
---	---	---	---
2.0	100.0	random_plus_manual_model_18	0.6519077
2.0	10.0	random_plus_manual_model_29	0.6565933
1.0	10.0	random_plus_manual_model_25	0.6648877
1.0	50.0	random_plus_manual_model_9	0.6668661
1.0	30.0	random_plus_manual_model_27	0.6670662
1.0	100.0	random_plus_manual_model_13	0.6676622
1.0	100.0	random_plus_manual_model_31	0.6684462
1.0	10.0	random_plus_manual_model_1	0.6687254
1.0	50.0	random_plus_manual_model_32	0.6689899
1.0	30.0	random_plus_manual_model_5	0.6696191

[32 rows x 5 columns]

In [ ]:
```
Rand_forest_model_rand = model_rand_search.get_grid(sort_by='auc', decreasing
Rand_forest_model_rand
```

Out[ ]:

Hyper-Parameter Search Summary: ordered by decreasing auc

max_depth	ntrees	model_ids	auc
6.0	100.0	random_plus_manual_model_16	0.7356001
6.0	100.0	random_plus_manual_model_22	0.7345767
6.0	50.0	random_plus_manual_model_12	0.7344319
6.0	50.0	random_plus_manual_model_28	0.7336739
6.0	30.0	random_plus_manual_model_20	0.7332505
6.0	30.0	random_plus_manual_model_8	0.7330098
6.0	10.0	random_plus_manual_model_4	0.7290180
6.0	10.0	random_plus_manual_model_23	0.7283533
4.0	100.0	random_plus_manual_model_15	0.7199165
4.0	30.0	random_plus_manual_model_26	0.7187054

	max_depth	ntrees	model_ids	auc
	2.0	10.0	random_plus_manual_model_2	0.6750081
	2.0	10.0	random_plus_manual_model_29	0.6678113
	1.0	100.0	random_plus_manual_model_13	0.6639136
	1.0	30.0	random_plus_manual_model_5	0.6636095
	1.0	50.0	random_plus_manual_model_32	0.6624400
	1.0	100.0	random_plus_manual_model_31	0.6610856
	1.0	50.0	random_plus_manual_model_9	0.6601883
	1.0	30.0	random_plus_manual_model_27	0.6567585
	1.0	10.0	random_plus_manual_model_1	0.6531592
	1.0	10.0	random_plus_manual_model_25	0.6508009

[32 rows x 5 columns]

In [ ]:
```
best_model = Rand_forest_model_rand.models[0]

# Now let's evaluate the model performance on a test set
# so we get an honest estimate of top model performance
best_model_perf1 = best_model.model_performance(valid)

best_model_perf1.auc()
```

Out[ ]:
```
0.7341018483113351
```

1. H2O AutoML

(a) Now using H20's AutoML find the best deep learning model for the same classification task. Use H2OAutoML and test a maximum of 20 models to find the best performing model. (2)

In [ ]:
```
aml = H2OAutoML(max_models=20, seed=1)
aml.train(x=predictors, y=response, training_frame=train)
```

```
AutoML progress: |
```

```
| (done) 100%
```

Out[ ]:
```
Model Details
=============
H2OStackedEnsembleEstimator : Stacked Ensemble
Model Key: StackedEnsemble_AllModels_1_AutoML_1_20221210_223053

No summary for this model

ModelMetricsBinomialGLM: stackedensemble
** Reported on train data. **

MSE: 0.1717606700684451
RMSE: 0.41444018877088296
LogLoss: 0.5168880107499636
AUC: 0.8290670184280022
AUCPR: 0.848251322307539
Gini: 0.6581340368560045
```

```
Null degrees of freedom: 10067
Residual degrees of freedom: 10056
Null deviance: 13925.936362522776
Residual deviance: 10408.056984461267
AIC: 10432.056984461267
```

Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.43005511039851657

	NO	YES	Error	Rate
NO	2914.0	1837.0	0.3867	(1837.0/4751.0)
YES	807.0	4510.0	0.1518	(807.0/5317.0)
Total	3721.0	6347.0	0.2626	(2644.0/10068.0)

Maximum Metrics: Maximum metrics at their respective thresholds

metric	threshold	value	idx
max f1	0.4300551	0.7733196	247.0
max f2	0.2343756	0.8661535	335.0
max f0point5	0.6048707	0.7758824	164.0
max accuracy	0.5100602	0.7472189	210.0
max precision	0.9948130	1.0	0.0
max recall	0.0760402	1.0	396.0
max specificity	0.9948130	1.0	0.0
max absolute_mcc	0.5100602	0.4933301	210.0
max min_per_class_accuracy	0.5144150	0.7457377	208.0
max mean_per_class_accuracy	0.5100602	0.7468488	210.0
max tns	0.9948130	4751.0	0.0
max fns	0.9948130	5300.0	0.0
max fps	0.0562889	4751.0	399.0
max tps	0.0760402	5317.0	396.0
max tnr	0.9948130	1.0	0.0
max fnr	0.9948130	0.9968027	0.0
max fpr	0.0562889	1.0	399.0
max tpr	0.0760402	1.0	396.0

Gains/Lift Table: Avg response rate: 52.81 %, avg score: 52.77 %

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
1	0.0101311	0.9626828	1.8935490	1.8935490	1.0
2	0.0200636	0.9404155	1.8746135	1.8841750	0.99
3	0.0302940	0.9234638	1.8751650	1.8811323	0.9902913
4	0.0400278	0.9118202	1.8549051	1.8747545	0.9795918

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
5	0.0500596	0.8994367	1.8560530	1.8710067	0.9801980
6	0.1000199	0.8408586	1.7806137	1.8258551	0.9403579
7	0.1500795	0.7955322	1.7207251	1.7907886	0.9087302
8	0.2000397	0.7554312	1.6187397	1.7478191	0.8548708
9	0.3001589	0.6758510	1.4633677	1.6529392	0.7728175
10	0.3999801	0.5990001	1.2849755	1.5611082	0.6786070
11	0.5000993	0.5235996	1.0839065	1.4655731	0.5724206
12	0.6000199	0.4499385	0.8959536	1.3707151	0.4731610
13	0.6999404	0.3753946	0.7171393	1.2774134	0.3787276
14	0.8000596	0.3019005	0.5466496	1.1859658	0.2886905
15	0.8999801	0.2185751	0.3783333	1.0962982	0.1998012
16	1.0	0.0551549	0.1335074	1.0	0.0705065

```
ModelMetricsBinomialGLM: stackedensemble
** Reported on cross-validation data. **

MSE: 0.20020711188685164
RMSE: 0.4474450937119007
LogLoss: 0.5850165961357272
AUC: 0.7566215079521776
AUCPR: 0.7701117638031549
Gini: 0.5132430159043553
Null degrees of freedom: 35250
Residual degrees of freedom: 35238
Null deviance: 48784.7107509362
Residual deviance: 41244.84006076104
AIC: 41270.84006076104
```

Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.34523525089372625

	NO	YES	Error	Rate
NO	6713.0	10053.0	0.5996	(10053.0/16766.0)
YES	2080.0	16405.0	0.1125	(2080.0/18485.0)
Total	8793.0	26458.0	0.3442	(12133.0/35251.0)

Maximum Metrics: Maximum metrics at their respective thresholds

	metric	threshold	value	idx
	max f1	0.3452353	0.7300358	289.0
	max f2	0.1873452	0.8480016	362.0
	max f0point5	0.5707982	0.7107568	178.0
	max accuracy	0.4954797	0.6928598	214.0
	max precision	0.9897326	1.0	0.0

metric	threshold	value	idx
max recall	0.0642784	1.0	399.0
max specificity	0.9897326	1.0	0.0
max absolute_mcc	0.4995609	0.3837273	212.0
max min_per_class_accuracy	0.5139708	0.6898485	205.0
max mean_per_class_accuracy	0.4995609	0.6917262	212.0
max tns	0.9897326	16766.0	0.0
max fns	0.9897326	18475.0	0.0
max fps	0.0642784	16766.0	399.0
max tps	0.0642784	18485.0	399.0
max tnr	0.9897326	1.0	0.0
max fnr	0.9897326	0.9994590	0.0
max fpr	0.0642784	1.0	399.0
max tpr	0.0642784	1.0	399.0

Gains/Lift Table: Avg response rate: 52.44 %, avg score: 52.44 %

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
1	0.0100139	0.9355643	1.8529829	1.8529829	0.9716714
2	0.0200278	0.9161459	1.7125235	1.7827532	0.8980170
3	0.0300133	0.9005440	1.7011357	1.7555988	0.8920455
4	0.0400272	0.887757	1.7827532	1.7623922	0.9348442
5	0.0500128	0.8759853	1.7282239	1.7555702	0.90625
6	0.1000255	0.8235680	1.6300950	1.6928326	0.8547930
7	0.1500099	0.778288	1.5487657	1.6448285	0.8121453
8	0.2000227	0.7385616	1.4170037	1.5878642	0.7430516
9	0.3000199	0.6653623	1.3486993	1.5081501	0.7072340
10	0.4000170	0.5917913	1.2221066	1.4366443	0.6408511
11	0.5000142	0.5203791	1.0630542	1.3619305	0.5574468
12	0.6000113	0.4494794	0.9342975	1.2906617	0.4899291
13	0.7000085	0.3802772	0.7709172	1.2164155	0.4042553
14	0.8000340	0.3079363	0.6619895	1.1470975	0.3471356
15	0.9000028	0.2316411	0.4908213	1.0742009	0.2573780
16	1.0	0.0608561	0.3321706	1.0	0.1741844

Cross-Validation Metrics Summary:

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid
accuracy	0.6638700	0.0128208	0.6778648	0.6486525	0.6627435	0.6544139

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid
auc	0.7567131	0.0021084	0.7547129	0.7583486	0.7570954	0.7543632
err	0.3361300	0.0128208	0.3221352	0.3513475	0.3372565	0.3455861
err_count	2369.8	92.166695	2263.0	2464.0	2407.0	2435.0
f0point5	0.6670924	0.0121398	0.6802884	0.6521052	0.6655973	0.6591672
f1	0.7313510	0.0035833	0.7334197	0.7280954	0.7363347	0.7279633
f2	0.8097621	0.0151452	0.7955533	0.8241319	0.8238957	0.8127931
lift_top_group	1.8426015	0.0596877	1.9027628	1.8396811	1.8737687	1.7448102
logloss	0.5850498	0.0021363	0.5865557	0.5833084	0.5847591	0.5878360
max_per_class_error	0.565955	0.0570027	0.5052505	0.6281975	0.5950902	0.5959988
---	---	---	---	---	---	---
mean_per_class_error	0.3467400	0.0141955	0.3310381	0.3623047	0.3501276	0.3573719
mse	0.2002199	0.0008122	0.2009359	0.1996136	0.2000669	0.2011726
null_deviance	9756.942	67.30502	9720.424	9710.795	9874.431	9750.64
pr_auc	0.7702631	0.0018805	0.7704577	0.7699251	0.7724190	0.7673437
precision	0.6302993	0.0172204	0.6489472	0.609684	0.6255351	0.6200990
r2	0.1971995	0.0033970	0.1941518	0.2001876	0.1975169	0.1933419
recall	0.8724751	0.0295358	0.8431745	0.9035881	0.8948349	0.8812551
residual_deviance	8249.45	67.87318	8241.107	8181.4834	8346.852	8283.784
rmse	0.4474587	0.0009074	0.4482588	0.4467813	0.4472884	0.4485227
specificity	0.434045	0.0570027	0.4947495	0.3718025	0.4049098	0.4040012

```
[22 rows x 8 columns]
```

```
[tips]
Use `model.explain()` to inspect the model.
--
Use `h2o.display.toggle_user_tips()` to switch on/off this section.
```

(b) Display the leaderboard, and identify the best performing model using it and print its parameters. (2)

In [ ]:
```
lb = aml.leaderboard
lb.head(rows=lb.nrows)
```

Out[ ]:

model_id	auc	logloss	aucpr	mea
StackedEnsemble_AllModels_1_AutoML_1_20221210_223053	0.756622	0.585017	0.770112	
StackedEnsemble_BestOfFamily_1_AutoML_1_20221210_223053	0.756361	0.58524	0.769764	
GBM_1_AutoML_1_20221210_223053	0.751977	0.589743	0.76388	
XGBoost_grid_1_AutoML_1_20221210_223053_model_2	0.750821	0.589996	0.765224	
GBM_2_AutoML_1_20221210_223053	0.75009	0.591395	0.759712	
GBM_3_AutoML_1_20221210_223053	0.748914	0.592695	0.75858	
GBM_4_AutoML_1_20221210_223053	0.747819	0.594896	0.758901	

model_id	auc	logloss	aucpr	mea
DeepLearning_grid_1_AutoML_1_20221210_223053_model_1	0.746329	0.601509	0.760493	
GBM_5_AutoML_1_20221210_223053	0.745466	0.595303	0.754611	
GBM_grid_1_AutoML_1_20221210_223053_model_1	0.744993	0.596193	0.753897	
XGBoost_2_AutoML_1_20221210_223053	0.743737	0.596246	0.758226	
XGBoost_grid_1_AutoML_1_20221210_223053_model_3	0.743353	0.59799	0.757639	
XGBoost_grid_1_AutoML_1_20221210_223053_model_1	0.742508	0.597333	0.757169	
XRT_1_AutoML_1_20221210_223053	0.742349	0.602508	0.75133	
XGBoost_3_AutoML_1_20221210_223053	0.740833	0.599946	0.755263	
DeepLearning_grid_2_AutoML_1_20221210_223053_model_1	0.74042	0.601622	0.753521	
GBM_grid_1_AutoML_1_20221210_223053_model_2	0.740006	0.599591	0.751981	
XGBoost_1_AutoML_1_20221210_223053	0.737746	0.60103	0.75346	
DeepLearning_grid_3_AutoML_1_20221210_223053_model_1	0.737743	0.604117	0.752804	
DRF_1_AutoML_1_20221210_223053	0.73602	0.618867	0.746985	
GLM_1_AutoML_1_20221210_223053	0.735356	0.604103	0.748015	
DeepLearning_1_AutoML_1_20221210_223053	0.732234	0.605814	0.746061	

[22 rows x 7 columns]

(c) Display the AUC score of the best model for the test set. (2)

In [ ]:
```
aml.leader
preds = aml.leader.predict(valid)
```

```
stackedensemble prediction progress: |████████████████████████████████
████| (done) 100%
```

In [ ]:
```
preds
```

Out[ ]:

predict	NO	YES
YES	0.276975	0.723025
YES	0.276975	0.723025
YES	0.261948	0.738052
YES	0.302014	0.697986
YES	0.312419	0.687581
YES	0.303859	0.696141
YES	0.252528	0.747472
YES	0.261948	0.738052
YES	0.303859	0.696141
YES	0.311557	0.688443

[8727 rows x 3 columns]

(d) Identify the best XGBoost model among all the models tested ranked by log loss as the criteria. (2)

In [ ]:
```
xgb = aml.get_best_model(algorithm="xgboost", criterion="logloss")
xgb
```

Out[ ]:
```
Model Details
=============
H2OXGBoostEstimator : XGBoost
Model Key: XGBoost_grid_1_AutoML_1_20221210_223053_model_2
```

Model Summary:

number_of_trees
49.0

```
ModelMetricsBinomial: xgboost
** Reported on train data. **

MSE: 0.1863920955244808
RMSE: 0.4317315086074687
LogLoss: 0.5525219214938155
Mean Per-Class Error: 0.30960708636897366
AUC: 0.7936144710605667
AUCPR: 0.8071367850319533
Gini: 0.5872289421211334
```

Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.3896628941098849

	NO	YES	Error	Rate
NO	8591.0	8175.0	0.4876	(8175.0/16766.0)
YES	2433.0	16052.0	0.1316	(2433.0/18485.0)
Total	11024.0	24227.0	0.3009	(10608.0/35251.0)

Maximum Metrics: Maximum metrics at their respective thresholds

metric	threshold	value	idx
max f1	0.3896629	0.7516389	268.0
max f2	0.2169247	0.8552953	350.0
max f0point5	0.5930708	0.7411929	171.0
max accuracy	0.4859416	0.7206320	224.0
max precision	0.9836319	1.0	0.0
max recall	0.0863908	1.0	396.0
max specificity	0.9836319	1.0	0.0
max absolute_mcc	0.4926000	0.4391631	221.0
max min_per_class_accuracy	0.5149050	0.7182039	209.0
max mean_per_class_accuracy	0.5121287	0.7192232	211.0

	metric	threshold	value	idx
	max tns	0.9836319	16766.0	0.0
	max fns	0.9836319	18473.0	0.0
	max fps	0.0654377	16766.0	399.0
	max tps	0.0863908	18485.0	396.0
	max tnr	0.9836319	1.0	0.0
	max fnr	0.9836319	0.9993508	0.0
	max fpr	0.0654377	1.0	399.0
	max tpr	0.0863908	1.0	396.0

Gains/Lift Table: Avg response rate: 52.44 %, avg score: 52.41 %

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
1	0.0100706	0.9373950	1.8640309	1.8640309	0.9774648
2	0.0200278	0.9216945	1.8418089	1.8529829	0.9658120
3	0.0300133	0.9092994	1.8257412	1.8439195	0.9573864
4	0.0402542	0.8958791	1.8013544	1.8330907	0.9445983
5	0.0500411	0.8848778	1.7356515	1.8140337	0.9101449
6	0.1000539	0.8238376	1.7285281	1.7712931	0.9064095
7	0.1500383	0.7799154	1.5898929	1.7108606	0.8337117
8	0.2000227	0.7360392	1.5379427	1.6676496	0.8064699
9	0.3001901	0.6602917	1.4101365	1.5817226	0.7394506
10	0.4001305	0.5936489	1.2596089	1.5012684	0.6605166
11	0.5000709	0.5218992	1.0739425	1.4158663	0.5631564
12	0.6000681	0.4481254	0.9229366	1.3337230	0.4839716
13	0.7000085	0.3799202	0.7708135	1.2533562	0.4042010
14	0.8000057	0.3111933	0.5756182	1.1686420	0.3018440
15	0.9000028	0.2302877	0.4203527	1.0855013	0.2204255
16	1.0	0.0564038	0.2304637	1.0	0.1208511

```
ModelMetricsBinomial: xgboost
** Reported on cross-validation data. **

MSE: 0.20235762997425993
RMSE: 0.44984178326858426
LogLoss: 0.5899957250481325
Mean Per-Class Error: 0.3319834898422497
AUC: 0.7508206550145875
AUCPR: 0.765224046569884
Gini: 0.501641310029175
```

Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.4052806258201599

	NO	YES	Error	Rate
NO	8569.0	8197.0	0.4889	(8197.0/16766.0)
YES	3236.0	15249.0	0.1751	(3236.0/18485.0)
Total	11805.0	23446.0	0.3243	(11433.0/35251.0)

Maximum Metrics: Maximum metrics at their respective thresholds

metric	threshold	value	idx
max f1	0.4052806	0.7273378	258.0
max f2	0.1712606	0.8477233	368.0
max f0point5	0.5706817	0.7066001	177.0
max accuracy	0.5167771	0.6880940	203.0
max precision	0.9820239	1.0	0.0
max recall	0.0650995	1.0	399.0
max specificity	0.9820239	1.0	0.0
max absolute_mcc	0.5167771	0.3756327	203.0
max min_per_class_accuracy	0.5186639	0.6869895	202.0
max mean_per_class_accuracy	0.5167771	0.6879925	203.0
max tns	0.9820239	16766.0	0.0
max fns	0.9820239	18465.0	0.0
max fps	0.0650995	16766.0	399.0
max tps	0.0650995	18485.0	399.0
max tnr	0.9820239	1.0	0.0
max fnr	0.9820239	0.9989180	0.0
max fpr	0.0650995	1.0	399.0
max tpr	0.0650995	1.0	399.0

Gains/Lift Table: Avg response rate: 52.44 %, avg score: 52.41 %

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
1	0.0100139	0.9350260	1.8313737	1.8313737	0.9603399
2	0.0201129	0.9180603	1.7570165	1.7940378	0.9213483
3	0.0300133	0.9045970	1.7540081	1.7808333	0.9197708
4	0.0400840	0.8924416	1.7297347	1.7679954	0.9070423
5	0.0500128	0.8796459	1.6945108	1.7534068	0.8885714
6	0.1000255	0.8271774	1.6138698	1.6836383	0.8462847
7	0.1500099	0.7795326	1.5152145	1.6275183	0.7945516
8	0.2000227	0.7383962	1.4321472	1.5786686	0.7509926

group	cumulative_data_fraction	lower_threshold	lift	cumulative_lift	response_rate
9	0.3000199	0.6628667	1.3378794	1.4984131	0.7015603
10	0.4000170	0.5935174	1.2091228	1.4260957	0.6340426
11	0.5000142	0.5236728	1.0825300	1.3573864	0.5676596
12	0.6000113	0.4525166	0.9207727	1.2846209	0.4828369
13	0.7000085	0.3791772	0.7774091	1.2121650	0.4076596
14	0.8000057	0.3068113	0.6513574	1.1420666	0.3415603
15	0.9000028	0.2281067	0.5101578	1.0718567	0.2675177
16	1.0	0.0608803	0.3532694	1.0	0.1852482

Cross-Validation Metrics Summary:

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_va
accuracy	0.6698533	0.0132182	0.6696922	0.6819858	0.6479433	0.67148
auc	0.7509419	0.0053686	0.7575662	0.7523400	0.7435789	0.74787
err	0.3301467	0.0132182	0.3303078	0.3180142	0.3520567	0.32851
err_count	2327.6	93.18959	2329.0	2242.0	2482.0	231(
f0point5	0.6734301	0.0137184	0.6749352	0.6886737	0.6511703	0.67425
f1	0.7283886	0.0055993	0.7348286	0.7327134	0.7215616	0.72408
f2	0.7935213	0.0131564	0.8063871	0.7827704	0.8090159	0.78187
lift_top_group	1.8373647	0.0363594	1.7781295	1.8550051	1.8750434	1.83481
logloss	0.5899959	0.0046970	0.584549	0.5879878	0.5963158	0.5932(
max_per_class_error	0.5220656	0.0525591	0.5482019	0.4747199	0.6018845	0.49703
mcc	0.3487203	0.0182069	0.3473044	0.3634108	0.3194323	0.34891
mean_per_class_accuracy	0.6610447	0.0137986	0.6570856	0.6727014	0.6391234	0.6643(
mean_per_class_error	0.3389553	0.0137986	0.3429144	0.3272986	0.3608766	0.3356(
mse	0.2023577	0.0020766	0.1998444	0.2016007	0.2051895	0.20363
pr_auc	0.7653908	0.0083804	0.7759413	0.7699819	0.7535828	0.7623(
precision	0.6412944	0.0186205	0.6401508	0.6621418	0.6114069	0.64467
r2	0.1885421	0.0076272	0.1975964	0.1903859	0.1781413	0.18387
recall	0.8441549	0.0258838	0.8623730	0.8201228	0.8801314	0.82581
rmse	0.4498371	0.0023069	0.4470396	0.4489997	0.4529785	0.45126
specificity	0.4779345	0.0525591	0.4517981	0.5252801	0.3981154	0.5029(

Scoring History:

timestamp	duration	number_of_trees	training_rmse	training_logloss	training_auc	train

timestamp	duration	number_of_trees	training_rmse	training_logloss	training_auc	train
2022-12-10 22:37:46	57.441 sec	0.0	0.5	0.6931472	0.5	
2022-12-10 22:37:47	58.689 sec	5.0	0.4539399	0.6003567	0.7565258	
2022-12-10 22:37:48	59.690 sec	10.0	0.4449813	0.5807043	0.7695196	
2022-12-10 22:37:49	1 min 0.156 sec	15.0	0.4419989	0.5742702	0.7750318	
2022-12-10 22:37:49	1 min 0.641 sec	20.0	0.4397077	0.5693799	0.7787136	
2022-12-10 22:37:50	1 min 1.070 sec	25.0	0.4376199	0.5650079	0.7826365	
2022-12-10 22:37:50	1 min 1.444 sec	30.0	0.4360995	0.5617895	0.7854689	
2022-12-10 22:37:51	1 min 1.871 sec	35.0	0.4346395	0.5586962	0.7883336	
2022-12-10 22:37:51	1 min 2.234 sec	40.0	0.4336088	0.5565347	0.7902789	
2022-12-10 22:37:51	1 min 2.668 sec	45.0	0.4327153	0.5546141	0.7918330	
2022-12-10 22:37:52	1 min 3.130 sec	49.0	0.4317315	0.5525219	0.7936145	

Variable Importances:

variable	relative_importance	scaled_importance	percentage
Distance	1944.7504883	1.0	0.1143620
Year.2008	1251.5601807	0.6435582	0.0735986
Year.2003	956.9573364	0.4920720	0.0562743
UniqueCarrier.HP	534.7208862	0.2749560	0.0314445
Year.2001	422.0031738	0.2169960	0.0248161
Year.2002	377.8424072	0.1942884	0.0222192
Year.1994	342.7400818	0.1762386	0.0201550
Year.2007	316.0342102	0.1625063	0.0185845
Year.2004	313.2542419	0.1610768	0.0184211
Year.1990	269.1669922	0.1384070	0.0158285

variable	relative_importance	scaled_importance	percentage
---	---	---	---
FlightNum.578	0.1500494	0.0000772	0.0000088
FlightNum.647	0.1460605	0.0000751	0.0000086
FlightNum.216	0.1420177	0.0000730	0.0000084
FlightNum.1425	0.0934580	0.0000481	0.0000055
FlightNum.2451	0.0788271	0.0000405	0.0000046
FlightNum.1066	0.0773363	0.0000398	0.0000045
FlightNum.109	0.0613793	0.0000316	0.0000036
Dest.KOA	0.0359911	0.0000185	0.0000021
FlightNum.1027	0.0307018	0.0000158	0.0000018
FlightNum.33	0.0130770	0.0000067	0.0000008

```
[664 rows x 4 columns]


[tips]
Use `model.explain()` to inspect the model.
--
Use `h2o.display.toggle_user_tips()` to switch on/off this section.
```