HeatWave AutoML

In-database machine learning with HeatWave

Purpose statement

This document provides an overview of features and enhancements included in HeatWave AutoML. It is intended solely to help you assess the benefits of HeatWave AutoML and to plan your I.T. projects.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Benchmark queries are derived from the TPC-H benchmark, but results are not comparable to published TPC-H benchmark results since they do not comply with the TPC-H specification.

Table of contents

Executive Summary

Artificial intelligence and machine learning (ML) have become pervasive technologies driving key features in enterprise as well as consumer products. End users have come to expect capabilities that leverage these technologies. The pace of innovation has given birth to multiple frameworks, techniques, and algorithms. This proliferation of ML techniques requires highly skilled data scientists and machine learning professionals who can apply them. Given the shortage of such professionals, it has become paramount to develop technologies that will enable citizen data scientists to leverage ongoing innovation in the machine learning field.

Emerging AI paradigms such as Generative AI and Vector Stores can complement traditional AI and ML. The AI platforms must support both traditional AI and ML as well as Generative AI so that customers can exploit synergy between them and create new classes of applications.

HeatWave uniquely integrates OLTP, OLAP, machine learning, Generative AI, Vector Store and Lakehouse in-database, enabling turnkey application development with enhanced performance and enterprise-grade data security. HeatWave users can combine machine learning and GenAI with other HeatWave's built-in capabilities, such as transaction processing, analytics across data warehouses and data lakes to create powerful and novel applications delivering more relevant and insightful responses—without the complexity, latency, risks, and cost of extract, transform, and load (ETL) duplication.

Current challenges of ML in databases

Developing and using machine-learning models requires skill sets in topics such as:

- Candidate algorithms/model types
- Hyperparameters that need to be tuned per algorithm
- Feature engineering
- Data preprocessing approach per data type
- Drift detection and retraining
- Knowledge of Python, as most ML algorithm frameworks are available only in Python

The current approach to use machine learning requires the user to perform ETL (Extract, Transform, Load) on the data stored in files and in databases. User must learn and use third-party tools and libraries to train a model and then perform inference and explanations. In addition to being onerous and time consuming, this process also has the potential to proliferate data outside of the database, causing data security and governance issues.

HeatWave AutoML

HeatWave AutoML enables users to train a model, generate inferences and explanations, on the data stored in object store as well as in MySQL database. It provides several advantages:

- **Fully Automated:** HeatWave AutoML fully automates the creation of tuned models, generating inferences and explanations, thus eliminating the need for the user to be an expert ML developer
- **SQL interface**: Provides the familiar MySQL interface for invoking machine learning capabilities
- **In-database machine learning**: Data and models always stay in-database resulting in a highly secured and efficient environment. No need to move the data using complex ETL processes and use of third party libraries is not needed making it an easy to use and manage environment.
- **Explanations**: All models created by HeatWave AutoML can be explained. Enterprises have a growing need to explain the predictions of machine learning models to build trust, demonstrate fairness, and comply with regulatory requirements.
- **Performance and Scalability**: The performance of HeatWave AutoML is much better at a lower cost than competing services such as Redshift ML. Furthermore, HeatWave AutoML scales with the size of the cluster.
- **Easy Upgrades**: HeatWave AutoML leverages state-of-the-art open-source Python ML packages that enable continual and swift uptake of newer (and improved) versions.

All these capabilities are available to HeatWave customers without any additional charge.

Building ML models on data stored in object store

HeatWave Lakehouse supercharges HeatWave AutoML by enabling machine learning operations – such as training, prediction, and explanation – to be performed on data stored in object store or inside the database. Now all machine learning tasks can be performed on collections of files that are easy to organize, visualize, and understand. This ability to use content from varied sources simplifies machine learning tasks, akin to the ease of working on the data warehouse.

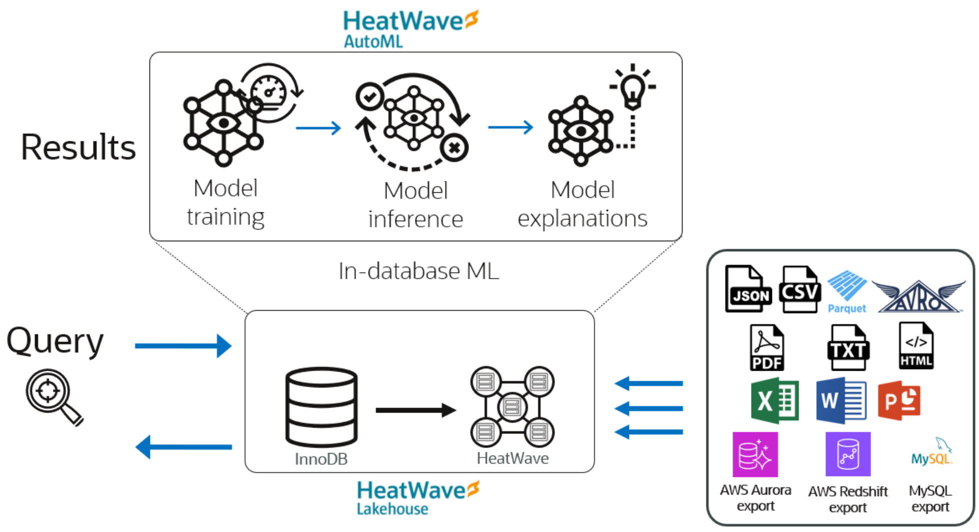
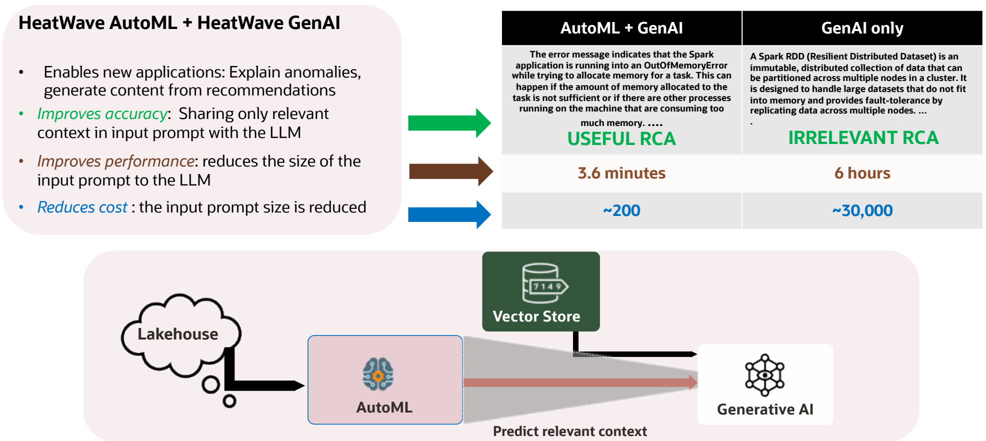


Figure 1 : Building ML models on data stored in object store and database

Synergy of built-in GenAI and ML

The combination of AutoML, GenAI, and vector store, all within the database, delivers more value to customers. It helps reduce costs and get more accurate results faster. For instance, AutoML excels at rapidly identifying hidden patterns in structured data and acts as a filter for data, which is then processed by GenAI.



Technology Background

HeatWave AutoML leverages Oracle AutoML [1], which automates the task of generating models. It replaces the laborious and time-consuming tasks that a data scientist typically performs, as listed below:

1. Preprocess the data
2. Select an algorithm from a set of algorithms to create a model
3. Select a suitable representative sample of data
4. Select only the relevant features to speed up the pipeline and reduce over fitting
5. Tune the hyperparameters
6. Ensure the model performs well on unseen data (also called generalization)

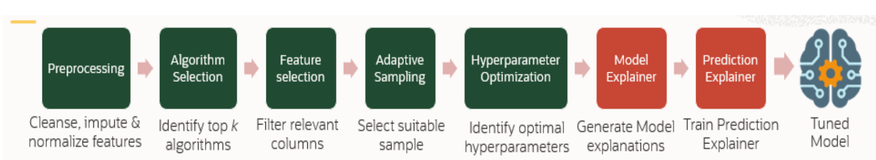


Figure 2: Machine learning pipeline automated by HeatWave AutoML

Oracle AutoML has a scalable design, minimizes the number of trials by extensive use of meta-learning, and provides an optimal model given a time budget. This proven technology has been integrated in various Oracle products, including the OCI Data Science Service and the Oracle Database.

Security benefits

HeatWave AutoML performs ML model training, inference and explanations on the data stored in object store and in MySQL database, without the data ever leaving HeatWave. All data and operations are executed in memory within the HeatWave cluster and the trained model is automatically stored in the MySQL database without any user data or model being transmitted to the client. The fully autonomous nature of the approach ensures that there is no human error that can cause security problems or errors in computation or data handling. This makes HeatWave AutoML far superior in terms of security to any existing ML solution for MySQL customers.

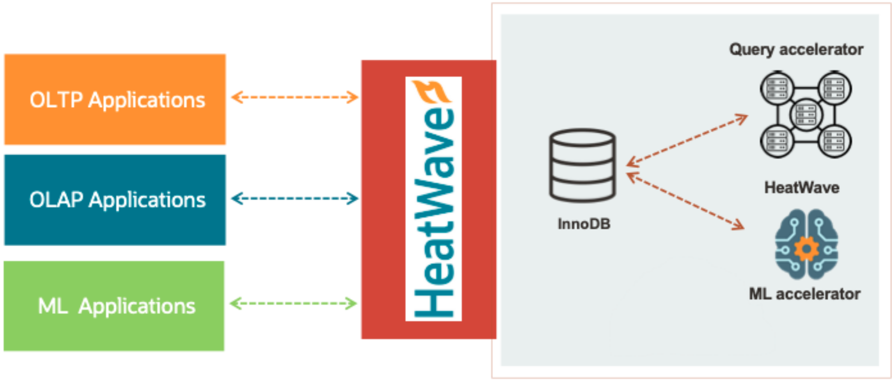


Figure 3: OLTP, OLAP, and ML workloads in a single database

Performance and Scalability

HeatWave AutoML is designed for high performance and scalability. High performance is achieved by the automated machine learning pipeline that consists of a novel non-iterative architecture comprised of multiple sequential stages. This design speeds up the pipeline as every stage's decision is made in a feed-forward manner. Key to the design is the reliance on *proxy models*—fast-performing models that are indicative of the performance of the final tuned model on a subset of a dataset. Furthermore, the algorithm selection stage at the beginning of the pipeline ensures that the downstream algorithm-dependent stages perform well without the need to iterate over multiple algorithms.

The HeatWave AutoML pipeline is designed in a flexible and highly parallel fashion, enabling us to distribute individual model fits and multiple parallel fits to all available compute nodes on a given HeatWave cluster. HeatWave AutoML has been optimized for both intra- and inter-model parallelism to achieve optimal performance on HeatWave cluster nodes. HeatWave AutoML can scale to dozens of HeatWave nodes (hundreds of cores), significantly reducing the ML training runtime as the cluster scales up. Furthermore, as training data size grows, user can scale up the cluster size to minimize the increase in training time.

Explainability

The integrated explainability module of HeatWave AutoML, helps users understand and interpret the model and its predictions.

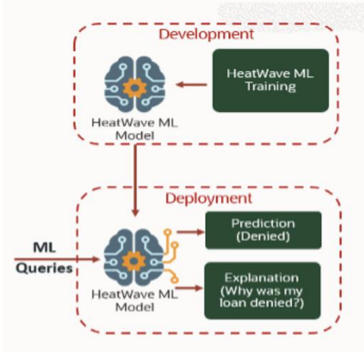


Figure 4: Model development and deployment

Deriving insights from the data and model helps the user answer questions around what factors matter most, why the model performs the way it does, and how it can be improved. Explainability can contribute to understanding the strengths and weaknesses in the user's data and within the predictions themselves.

Model management and use

Once HeatWave cluster has been provisioned and data is loaded into HeatWave, users can create the model, deploy the model, and use it to create predictions and explanations. Periodically, users will also check the model quality. If model drift is detected, users can check the results of model explanation and recreate the model on more recent data and new data features.

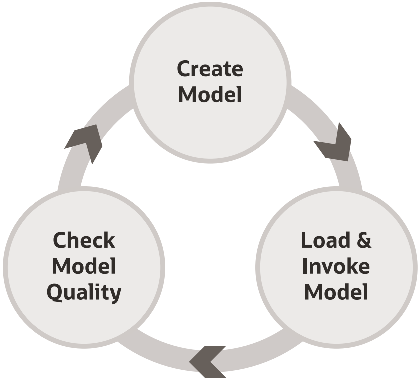


Figure 5: Model lifecycle

Create model

HeatWave AutoML supports multiple model types such as Classification, Regression, Time Series Forecasting, Anomaly Detection, Recommender System etc. Based on the problem at hand, users must select the appropriate model type.

HeatWave AutoML handles this data preprocessing as part of the model creation, however, users must conduct due diligence to provide a comprehensive dataset.

Once the key data attributes are identified, users needs to consolidate the data in a train table and invoke the sys.ML_TRAIN procedure to create a trained model as well as an optimized explainer model (which can provide insight into the model's behavior).

Example:

```
mysql> CALL sys.ML_TRAIN('mlcorpus.census_train', 'revenue',
JSON_OBJECT('task', 'classification'), @model);
mysql> CALL sys.ML_TRAIN('mlcorpus.boston_train', 'target',
JSON_OBJECT('task', 'regression'), @boston_model)
```

Model catalog

The model catalog is a table (MODEL_CATALOG) within the user schema (ML_SCHEMA_<current username>) created by ML_TRAIN if it does not already exist. The model catalog stores all models trained during ML_TRAIN and each model becomes a row in the MODEL_CATALOG table. The model catalog makes ML models first-class citizens of the database, enabling them to be backed up, restored, encrypted, and follow other DB procedures and protocols that regular DB tables provide. The catalog also helps with the sharing of models between multiple users as owners can control access and rights to their tables.

Load and invoke model

Load and unload model

The models stored in the model catalog must be loaded in memory before they can be used with the ML_MODEL_LOAD routine. A model remains loaded until it is unloaded using the ML_MODEL_UNLOAD routine or until the HeatWave AutoML driver is restarted. It is important to unload models that are not needed to run the HeatWave AutoML effectively to free up the memory.

Example: Load model

```
mysql> CALL sys.ML_MODEL_LOAD(@model, NULL);
Query OK, 0 rows affected (1.12 sec)
```

Example: Unload model

```
mysql> CALL sys.ML_MODEL_UNLOAD(@model);
Query OK, 0 rows affected (1.12 sec)
```

Prediction on a row

Users can predict the outcome for a specific row using the ML_PREDICT_ROW function. The sys.ML_PREDICT_ROW is a stored function that runs in-line inference on a single row of data using a previously trained model. The user provides the input row of data in the JSON format, for which the prediction is performed using the trained model object.

Example: Predict row

```
mysql> SELECT sys.ML_PREDICT_ROW('{"index": 1,"age":
38,"workclass": "Private","fnlwgt": 89814,"education": "HS-
grad","education-num": 9,"marital-status": "Married-civ-
spouse","occupation": "Farming-fishing","relationship":
"Husband","race": "White","sex": "Male","capital-gain":
0,"capital-loss": 0,"hours-per-week": 50,"native-country":
"United-States"}', @model);
|
+-------------------------------------------------------+
```

```
| {"age": 38, "sex": "Male", "race": "White", "index": 1,
"fnlwgt": 89814, "education": "HS-grad", "workclass": "Private",
"Prediction": "<=50K", "occupation": "Farming-fishing",
"capital-gain": 0, "capital-loss": 0, "relationship": "Husband",
"education-num": 9, "hours-per-week": 50, "marital-status":
"Married-civ-spouse", "native-country": "United-States"} |
+--------------------------------------------------------+
1 row in set (2.36 sec)
```

Prediction on a table

Users can create predictions for an entire table using the ML_PREDICT_TABLE. The sys.ML_PREDICT_TABLE creates and populates a new table with features and predictions for each row of the input table. Predictions across rows are done in parallel.

Example: Prediction on a table

```
mysql> CALL sys.ML_PREDICT_TABLE('mlcorpus.census_test_temp',
@model, 'mlcorpus.census_predictions');

Query OK, 0 rows affected (4.54 sec)

mysql> SELECT `index`, `education-num` AS education_level, `hours-
per-week` AS hours_per_week, Prediction FROM census_predictions;

+-------+-----------------+----------------+------------+

| index | education_level | hours_per_week | Prediction |

+-------+-----------------+----------------+------------+

|     0 |               7 |             40 | <=50K      |

|     1 |               9 |             50 | <=50K      |

|     2 |              12 |             40 | <=50K      |

|     3 |              10 |             40 | >50K       |

|     4 |              10 |             30 | <=50K      |

+-------+-----------------+----------------+------------+

5 rows in set (0.00 sec)
```

Explain predictions on a row

Users can explain predictions for a specific row using the ML_EXPLAIN_ROW function. The sys.ML_EXPLAIN_ROW is a stored function that provides the user with an interface to create in-line explanations from a single row of input data. Explanations help the user perform knowledge discovery by explaining which features matter most to the model (captured during ML_TRAIN), and which features contribute the most to individual predictions (via ML_EXPLAIN).

Example: One row input

```
mysql> SELECT sys.ML_EXPLAIN_ROW('{"index": 1,"age": 38,"workclass":
"Private","fnlwgt": 89814,"education": "HS-grad","education-num":
```

```
 9,"marital-status": "Married-civ-spouse","occupation": "Farming-
fishing","relationship": "Husband","race": "White","sex":
"Male","capital-gain": 0,"capital-loss": 0,"hours-per-week":
50,"native-country": "United-States"}', @model);
|
| {"age": 38, "sex": "Male", "race": "White", "index": 1, "fnlwgt":
89814, "education": "HS-grad", "workclass": "Private", "Prediction":
"<=50K", "occupation": "Farming-fishing", "capital-gain": 0,
"capital-loss": 0, "relationship": "Husband", "education-num": 9,
…
"capital-loss_attribution": 0.0, "relationship_attribution": 0.0928,
"education-num_attribution": 0.1305, "hours-per-week_attribution":
0.1806, "marital-status_attribution": 0.0676, "native-
country_attribution": 0.0001} |
+----------------------------------------------------------
1 row in set (4.41 sec)
```

Explain prediction on a table

The sys.ML_EXPLAIN_TABLE creates and populates a new table with features, predictions, and explanations for each row of the input table. Explanations across rows are done in parallel. The loaded model's training columns must match the ML_EXPLAIN_TABLE input columns.

Example: Explain prediction on a table

```
mysql> CALL sys.ML_EXPLAIN_TABLE('mlcorpus_v4.census_test_naive',
@model, 'mlcorpus_v4.census_explanations');
Query OK, 0 rows affected (12.95 sec)

mysql> SELECT `index`, `education-num` AS education_level, `hours-
per-week` AS hours_per_week, Prediction, `education-num_attribution`
AS education_level_attr, `hours-per-week_attribution` AS
hours_per_week_attr FROM census_explanations;
+-------+-----------------+----------------+------------+----------------------+---------------------+
| index | education_level | hours_per_week | Prediction | education_level_attr | hours_per_week_attr |
+-------+-----------------+----------------+------------+----------------------+---------------------+
|     0 |               7 |             40 | <=50K      |               -0.001 |              -0.002 |
|     1 |               9 |             50 | <=50K      |              -0.1307 |             -0.1807 |
|     2 |              12 |             40 | <=50K      |              -0.2435 |             -0.2101 |
|     3 |              10 |             40 | >50K       |                0.007 |              0.0053 |
|     4 |              10 |             30 | <=50K      |               0.0007 |             -0.0002 |
+-------+-----------------+----------------+------------+----------------------+---------------------+
5 rows in set (0.00 sec)
```

Check model quality

The sys.ML_SCORE procedure computes the model quality by generating predictions on given test data and comparing it vs the ground truth labels. The ML_SCORE API requires a string argument that specifies the scoring metric to be used. HeatWave

AutoML supports multiple standard scoring metrics as described here for classification and regression.

Example:

```
mysql> CALL sys.ML_SCORE('mlcorpus_v4.census_test', 'revenue',
@model, 'balanced_accuracy', @score);
Query OK, 0 rows affected (5.34 sec)

mysql> SELECT @score;
+--------------------+
| @score             |
+--------------------+
| 0.7961280941963196 |
+--------------------+
1 row in set (0.00 sec)
```

Data drift detection

ML models can become outdated over time and lose their ability to predict accurately. This may happen due to data drift whereby distribution of input data changes overtime and model cannot predict accurately as it is trained on older data. It is important to be able to measure the data drift and then retrain the model as needed on the latest data so that model continues to predict accurately.

HeatWave AutoML provides functionality to detect data drift using a drift detector trained during model training. The metric such as mean and variance for the model features are computed during model training and stored in the model catalog. At the time of inference, a user can ask the trained detector to evaluate each sample's drift level. The drift value can be evaluated across all samples and features or for a specific feature. It reveals up to top three features with the highest feature drift. This information can be used to trigger model training if the samples have drifted beyond a preset threshold.

Time Series Forecasting

Time series forecasting involves using time ordered events from past as well as other variables to predict future values.

While analyzing time series, it becomes important to exploit temporal dependency and internal structure comprising of elements such as seasonality, trend, and residual. There are several time series forecasting algorithms, each best suited to a varying degree of strength of basic time-series characteristics. The choice of the optimal algorithm requires a statistician trained in time-series analysis for effective forecasting. Given the complexity involved, an automated approach for time series forecasting is highly desirable.

HeatWave AutoML offers a fully automated forecasting pipeline that can automatically preprocess, select the best algorithm, and tune its hyperparameters for a given time-series dataset resulting in unmatched model training performance and high forecasting accuracy.

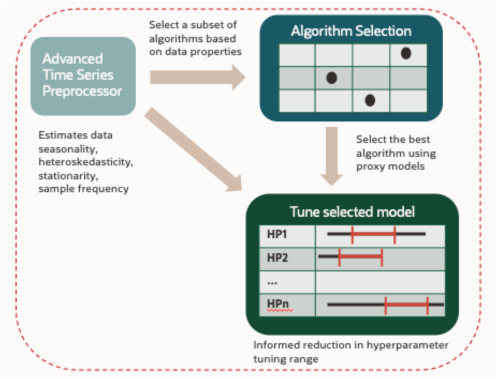


Figure 6 : HeatWave AutoML forecasting pipeline

The HeatWave AutoML automated forecasting pipeline uses a patented technique that consists of stages such as advanced time-series preprocessing, algorithm selection and hyper parameter tuning. The advanced time-series stage prunes the search space and estimates basic time-series characteristics (seasonality, trend etc.) and these estimates are used later by the algorithm selection and hyperparameter tuning stages. The algorithm selection stage estimates the best algorithm for a given time-series dataset from the set of supported algorithms. The hyperparameter tuning stage tunes the hyperparameters for the algorithm in a range suggested by the preprocessor. This results in significant speedup by reducing the number of trials and improves generalization of tuned models.

The user can get predictions and can specify a confidence level at which the upper and lower bounds of forecasted outcome can be generated.

Unsupervised Anomaly Detection

Anomaly detection is a technique for finding unusual patterns in data. It has found applications in a wide variety of fields, including fraud detection, network intrusion detection, detecting life-threatening medical conditions, quality control etc.

Anomaly detection is particularly challenging because of issues such as lack of labelled data, the need for different algorithms to address various types of anomalies, and the unbalanced nature of data given that, by definition, anomalies are rare.

HeatWave AutoML detects anomalies in unlabeled data using a novel and patented technique called Generalized kth Nearest Neighbors (GkNN) which is based on a single ensemble algorithm that does not require tuning of hyperparameters. It identifies common types of anomalies such as local, global, and clustered which typically require separate algorithms to detect. It provides high performance on the Unsupervised Anomaly Detection Benchmark (UADB) datasets, outperforming some of the most widely utilized algorithms such as k-th Nearest Neighbor (kNN) and Local Outlier Factor (LOF). In addition to GkNN, HeatWave AutoML supports two additional algorithms: Principal Component Analysis (PCA) and Generalized Local Outlier Factor (GLOF), a proprietary algorithm. These algorithms provide added ability to detect anomalies and user can provide options in ML_TRAIN procedure to use PCA and GLOF algorithms.

HeatWave AutoML supports anomaly detection in a fully automated way so that the user does not need to select a specific algorithm to address a particular type of anomaly. This drastically improves the performance given that HeatWave AutoML does not need to evaluate different types of algorithms based on the anomaly type, unlike other approaches to anomaly detection.

Anomaly Type/Technique	**HeatWave**	KNN	LoF
Local	✓	✓	✓
Global	✓	✓	✓
Cluster	✓	✓	✓

None of the competing products such as Google BigQuery ML, Redshift ML or Snowflake offers a fully automated solution for anomaly detection as HeatWave AutoML does. HeatWave is much faster and accurate compared to Redshift ML.

Recommender system

Recommender systems (also known as 'recommendation systems') are commonly used in e-commerce to recommend new products to users based on their prior history and preferences. The concept behind recommendation systems is finding patterns in consumer behavior to predict users' preferences, even before they have interacted with the product.

The HeatWave AutoML Recommender System leverages models based on collaborative filtering methods. These models are trained uniquely on past user-item interactions. It supports recommendations based on both explicit and implicit feedback.

- Explicit Feedback: If the data is composed of ratings directly provided by the users, then it is categorized as explicit feedback. The user ratings can be positive or negative. HeatWave AutoML use a variety of models for explicit feedback, including NormalPredictor, Baseline, Slopeone, CoClustering, SVD, SVDpp, and NMF.
- Implicit Feedback: If the data contains information produced from user behavior like clicks and purchases, this is considered implicit feedback. This type of data is more widespread, as the user does not have to explicitly express their taste about the item.

HeatWave AutoML supports the following types of recommendations:

- Items that the user will like
- Users who will like an item
- User ratings of an item
- Identify similar users
- Identify similar items

Track the progress of training, inference, and explanations

HeatWave AutoML progress tracking can be used to monitor the progress of training, inference, and explanations for HeatWave AutoML. The goal of the progress tracker is to provide visibility to the end user about the execution status of the HeatWave AutoML operations, for example, how much an operation has progressed, which stages have been completed, any error that has occurred during the operation, or whether the operation has been aborted.

The progress tracker can be invoked on HeatWave AutoML using SQL queries. To initiate progress tracking, the user needs to open two MySQL client terminals. The first terminal is used to start the machine learning query, while the second terminal is used to monitor the progress of the operation.

Interactive Console

The interactive console for HeatWave, and HeatWave AutoML, is an integrated environment that provides users the ability to manage the database schema objects, run interactive queries, monitor performance, and use machine learning capabilities such that a business analyst can easily develop applications, manage data objects, and machine learning models. Users can train machine learning models, score, and explain them, run predictions and What-If scenarios to view the impact of feature changes on model outcomes. The console is initially available for HeatWave on AWS.

Scenario analysis - Users can change the values of certain features of a data record and compare the model outcome with the original values (aka baseline).

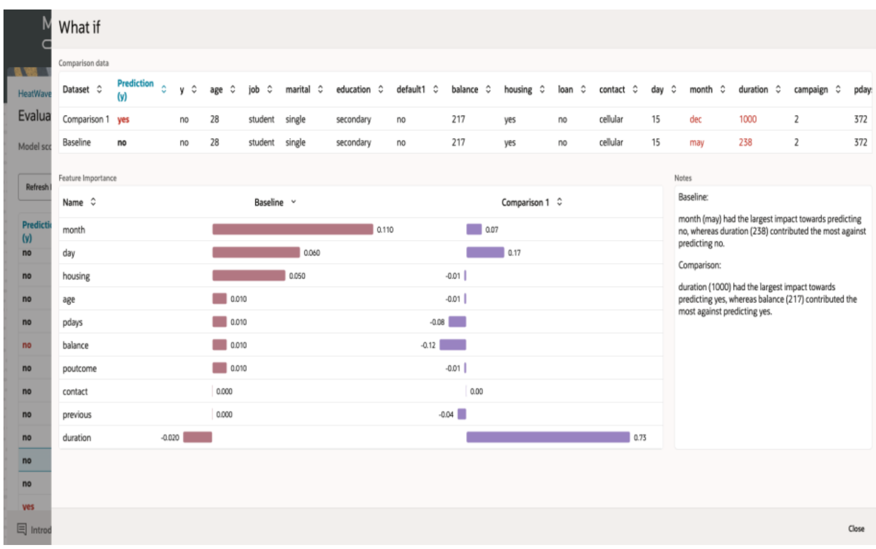


Figure 7 : What-if analysis using interactive console

Integration with interactive development tools

A user can easily connect to HeatWave from interactive notebook environments such as Jupyter and Apache Zeppelin notebooks and run transactional, analytics and machine learning queries. In addition, in the notebook environment, users can leverage various features available for numeric computations, data processing, data visualization and more using their language of choice.

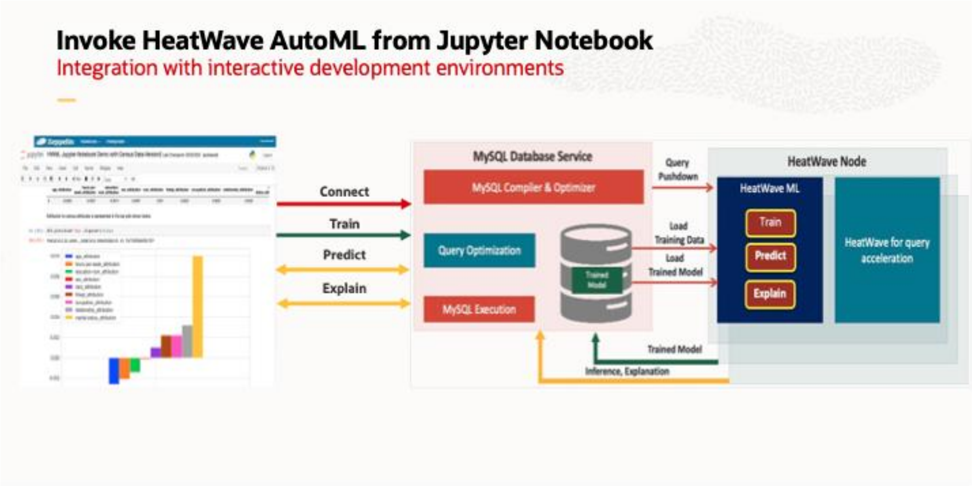


Figure 8: Invoke HeatWave AutoML from notebook environment

Performance comparison

We ran benchmarks on several datasets relevant to enterprise use cases, and compared the performance of training time, quality of models and scalability with Redshift ML. This was done for classification and regression datasets.

Classification comparison

The table below compares the balanced accuracy and training times of HeatWave AutoML vs. Redshift ML. In some cases, Redshift ML was around 200x slower than HeatWave AutoML. We used geometric mean for the comparison to dampen the effect of these outliers. The geometric mean average indicates that HeatWave AutoML's training time is ~25x faster than Redshift ML and has slightly better accuracy.

This significant improvement in performance enables users to retrain more frequently, and thereby keep models current with the data, which results in better prediction accuracy.

Dataset	Accuracy		Training Time (minutes)		Speedup	Cost ($)		Cheaper
	Redshift ML	HeatWave ML	Redshift ML (MAX_CELLS: 1M, MAX_RUNTIME: 5400s)	HeatWave ML (2 nodes)		Redshift ML 1 year plan	HeatWave ML	
Airlines	0.5	0.6524	90.00	2.71	33.21	6.23	0.0479	130.03
Bank	0.8378	0.7115	90.00	3.72	24.19	5.68	0.0658	86.30
CNAE-9	X	0.9167	X	5.91	X	X	0.1045	X
Connect-4	0.6752	0.6970	90.00	7.13	12.62	6.18	0.1261	49.05
Fashion MNIST	X	0.9073	X	181.85	X	X	3.2151	X
Nomao	0.9512	0.9602	90.00	3.30	27.27	5.96	0.0583	102.14
Numerai	0.5	0.5184	90.00	0.34	264.71	5.49	0.0060	913.49
Higgs	0.5	0.758	90.00	68.58	1.31	7.27	1.2125	5.99
Census	0.7985	0.7946	90.00	1.22	73.77	6.12	0.0216	283.95
Titanic	0.9571	0.7660	90.00	0.47	191.49	5.60	0.0083	674.32
CC Fraud	0.9154	0.9256	90.00	29.06	3.10	6.70	0.5138	13.03
KDD Cup	X	0.5	X	3.55	X	X	0.0628	X
GEOMEAN	**0.712**	**0.754**	**90.00**	**3.561**	**25.271**	**6.115**	**0.063**	**97.129**

Figure 9: Accuracy and training time comparison

Regression comparison

The table below compares the r2 values and training times with Redshift ML. Using the geometric mean average, the results indicate that HeatWave AutoML's training time is ~25x faster than Redshift ML for comparable accuracy.

Dataset	Accuracy		Training Time (minutes)		Speedup	Cost ($)		Cheaper
	Redshift ML	HeatWave ML	Redshift ML (MAX_CELLS: 1M, MAX_RUNTIME: 5400s)	HeatWave ML (2 nodes)		Redshift ML 1 year plan	HeatWave ML	
Black Friday	0.54	0.53	90.00	1.14	78.80	2.95	0.02	146.10
Diamonds	0.98	0.98	90.00	2.40	37.42	5.13	0.04	120.61
Mercedes	X	0.61	X	1.16	X	X	0.02	X
News Popularity	0.02	0.01	90.00	0.60	149.13	4.15	0.01	389.08
NYC_taxi	0.19	0.25	90.00	7.34	12.26	2.82	0.13	21.76
Twitter	0.88	0.93	90.00	44.24	2.03	3.64	0.78	4.66
GEOMEAN	**0.27**	**0.26**	**90.00**	**3.52**	**25.58**	**3.64**	**0.06**	**58.66**

Figure 10: Accuracy and training time comparison

Scalability comparison

The plot below shows the impact of adding more nodes to HeatWave AutoML (i.e. from 1 through 16 nodes) on total training time as well as the time consumed by key training pipeline components for the Higgs data set. There are two key takeaways from this chart:

- The most time consuming stages of the HeatWave AutoML training pipeline typically accelerate the most with a larger cluster.
- The total training time was reduced from 7108 seconds to 1208 seconds as the cluster size was increased from 1 to 16 nodes.

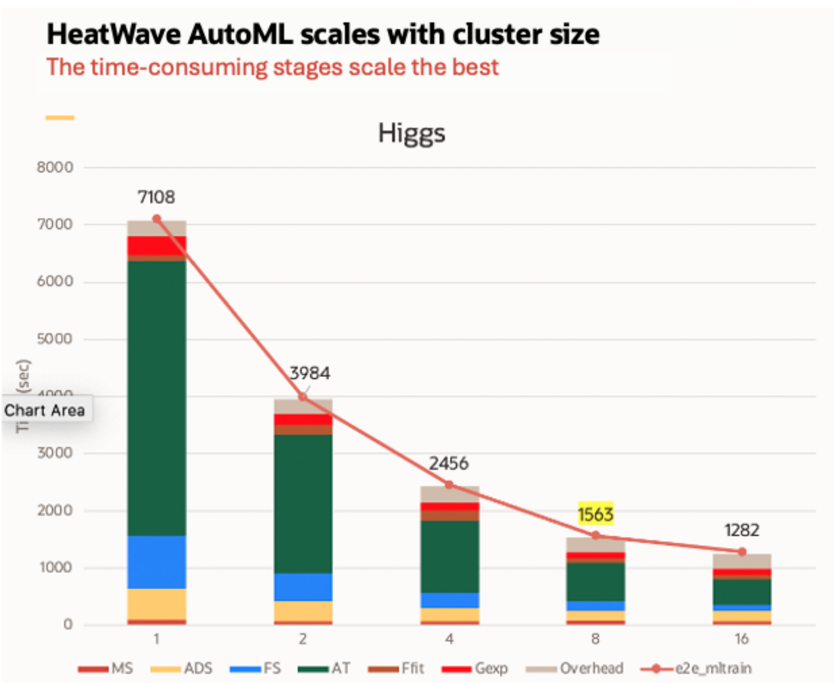


Figure 11: Impact of larger cluster on ML pipeline

- MS – Model Selection stage selects an algorithm
- ADS – Adaptive Data Sampling stage selects an optimal number of rows for the remainder of stages
- FS – Feature Selection stage selects the relevant subset of features
- AT – Autotune stage is the hyperparameter optimization stage that tunes selected algorithm's hyperparameters
- FFit – Final Fit stage fits the tuned algorithm on the full dataset that includes all rows
- Gexp – Global Explainer training stage explains the model
- Overhead – end-to-end ML_TRAIN overhead time includes transfer of the dataset to cluster, initialization, and completion time
- e2e_mltrain – end-to-end total ML_TRAIN time from MySQL client perspective.

Secondly, we compared the scalability of HeatWave AutoML with Redshift ML

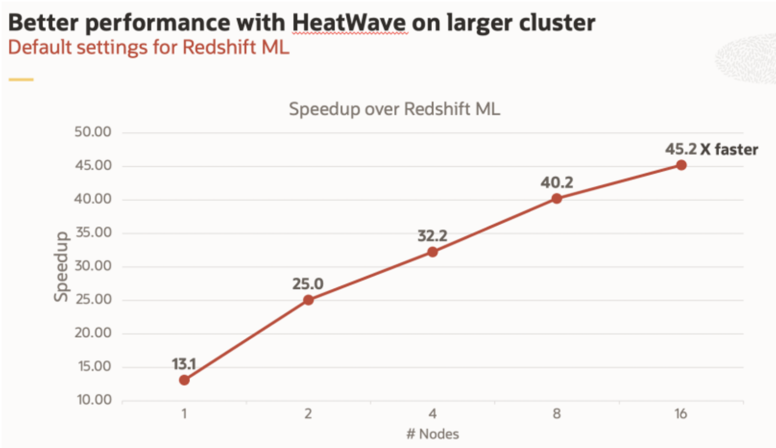


Figure 12: Training speedup comparison with larger clusters

Cost comparison

For customers of HeatWave, there is no additional cost for using HeatWave AutoML. Customers only pay for the provisioned cluster in contrasts with other services like Redshift ML where customers are charged for the use of SageMaker and S3 storage.

Note that for Redshift ML, we do not calculate the costs based on Redshift ML documentation [5] under "Redshift ML pricing" section, as the actual cost incurred during training are significantly different from the documented pricing. We calculated the cost based on the instance shape and runtime of the SageMaker instance that was invoked by Redshift ML.

Compared to Redshift ML, HeatWave AutoML is 97x cheaper in classification tests and 58.7x cheaper in regression tests.

Dataset	Cost $ Redshift ML 1 year plan (MAX_CELLS: 1M, MAX_RUNTIME: 5400s)	HeatWave ML (2 nodes)	Cheaper
Airlines	6.23	0.0479	130.03
Bank	5.68	0.0658	86.30
CNAE-9	X	0.1045	X
Connect-4	6.18	0.1261	49.05
Fashion MNIST	X	3.2151	X
Nomao	5.96	0.0583	102.14
Numerai	5.49	0.0060	913.49
Higgs	7.27	1.2125	5.99
Census	6.12	0.0216	283.95
Titanic	5.60	0.0083	674.32
CC Fraud	6.70	0.5138	13.03
KDD Cup	X	0.0628	X
GEOMEAN	6.115	0.063	97.129

Figure 13 : Cost comparison for classification models

Dataset	Cost $ Redshift ML 1 year plan (MAX_CELLS: 1M, MAX_RUNTIME: 5400s)	HeatWave ML (2 nodes)	Cheaper
Black Friday	2.95	0.02	146.10
Diamonds	5.13	0.04	120.61
Mercedes	X	0.02	X
News Popularity	4.15	0.01	389.08
NYC_taxi	2.82	0.13	21.76
Twitter	3.64	0.78	4.66
GEOMEAN	3.64	0.06	58.66

Figure 14 : Cost comparison for regression datasets

Conclusion

HeatWave supports OLTP, real-time analytics across data warehouse and data lake, machine learning, and generative AI with vector store in one fully managed cloud service, avoiding the complex, time-consuming, and expensive ETL to separate services. Customers can leverage the synergy between GenAI and ML to derive significant benefits, including reduced cost and improved performance and accuracy.

HeatWave AutoML, fully automates the creation of tuned ML models, generating inferences and explanations, thus eliminating the need for users to be an expert. Benchmarks demonstrate that, on average, HeatWave AutoML produces more accurate results than Amazon Redshift ML, trains models up to 25X faster at 1% of the cost, and scales as more nodes are added. HeatWave AutoML is available at no additional cost to HeatWave customers.

Resources

Learn more about HeatWave: https://www.oracle.com/heatwave

Try HeatWave for free; https://www.oracle.com/heatwave/free

References

[1] Anatoly Yakovlev, Hesam Fathi Moghadam, Ali Moharrer, Jingxiao Cai, Nikan Chavoshi, Venkatanathan Varadarajan, Sandeep R. Agrawal, Sam Idicula, Tomas Karnagel, Sanjay Jinturkar, and Nipun Agarwal. 2020. Oracle AutoML: a fast and predictive AutoML pipeline. <i>Proc. VLDB Endow.</i> 13, 12 (August 2020), 3166–3180. DOI:https://doi.org/10.14778/3415478.3415542

[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

[4] https://aws.amazon.com/savingsplans/ml-pricing/

[5] https://aws.amazon.com/redshift/pricing/

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle