Position Paper

Deploy Machine Learning for the New Speed and Scale of Business

Built into Vertica's core—no need to install separate packages—in-database machine learning supports the entire predictive analytics process with massively parallel processing and a familiar SQL or Python interface, allowing data scientists and analysts to use all their data to accelerate business outcomes without limits or compromises.

opentext™

Table of Contents

Vertica's in-database machine learning allows you to embrace the power of big data and accelerate business outcomes with no limits and no compromises.

Purpose of this Position Paper

With the ever-increasing volume, velocity, and variety of data, it's harder than ever for organizations to discover the critical hidden insights contained in their data, then create competitive differentiation to increase profitability. This position paper explores the new approach data scientists and BI analysts can take in leveraging data for game-changing insights, and it explains how in-database machine learning in OpenText™ Vertica eliminates the constraints of small-scale analytics when applied to extremely large or high velocity datasets for highest possible accuracy at lightning speeds. In addition, it explains how the unique benefits of Vertica's end-to-end in-data base machine learning both complements and extends the capabilities of traditional tools, including C++,Java, Python ,Jupyter, and R.

Machine Learning: a Competitive Advantage

"Vertica enables us to analyze very large sets of structured, semi- structured, and unstructured data in real time."

DR. JOCHEN SCHOLOSSER
Chief Strategy Officer
Adform

In today's data-driven world, creating a competitive advantage depends on your ability to transform massive volumes of data into meaningful insights rapidly, and act on those insights. Companies that use advanced analytics and machine learning are twice as likely to be top financial performers, and three times more likely to execute effective decisions.[1] 88% of companies expect to increase the speed of innovation as a result of their use of artificial intelligence (AI)and machine learning (ML).[2]

Unfortunately, the huge increase in velocity, volume, and variety of data has simultaneously increased the complexity of building predictive models. Too often, machine learning tools cannot process necessary data sets at the modern speed and scale of business, or they require a huge investment in skilled people's time to accomplish the simplest goals. Vertica's in-database machine learning allows you to accelerate business outcomes with no limits and no compromises.

What Is Machine Learning?

Machine learning is a mathematical way of identifying patterns and relationships in data, which can be used to predict new values. Machine learning provides in-depth understanding of many essential aspects of business. For example, a company can use patterns in customer behavior to personalize the consumer experience, prevent churn, detect fraud, and increase the bottom line.

This is creating a fundamental shift in the way businesses are operating—from being reactive to being proactive. Machine learning helps data scientists and analysts unlock valuable insights from huge, difficult-to-understand data sets. Those insights facilitate more informed decision making.

1 *"Creating Value through Advanced Analytics." Bain & Company, February 2015*
2 *EMA, June, 2020*—**www.vertica.com/resource/the-race-for-a-unified-analytics-warehouse/**

However, to capture the full value of all accessible data, we need to change the scale and speed through which data for ML algorithms is prepared, and the ML models are trained, deployed, and managed. Data volumes are far too large for many data science and machine learning tools to handle, and the relationships between disparate data sources—from back-end customer databases to click-stream behavior or sensor logs—are far too complex for ordinary business intelligence tools.

Vertica's in-database machine learning tools free data scientists from the constraints of traditional approaches and increase model accuracy by allowing data scientists to discover and display patterns buried in ever-larger data sets. More accurate predictions can be turned into better customer service, superior products, reduced downtime, increased profit margin, and an overall competitive advantage in many industries.

"With high speed analytics from Vertica, we can tune our website intelligently to stay ahead of other players in the market."

IDAN ZALZBERG
CTO
Agoda

Barriers to Applying Machine Learning at Scale

There are several challenges when it comes to applying machine learning to the massive volumes of data that organizations collect and store. Despite the complexity of predictive analytics, today's business leaders expect insights in ever shorter time frames. However, many machine learning algorithms were not designed for distributed processing, which is the only way to process more data faster. The algorithms, and the data exploration and preparation functions that support them, must be purpose-built to take advantage of modern distributed and parallel engines.

Traditionally, data scientists had to build and tune models using only small subsets of data (down-sampling). This often results in inaccuracies, delays, increased costs to deploy, and generally slower access to critical insights:

- **Slower development:** Delays in moving large volumes of data between systems increases the amount of time data scientists spend creating predictive analytics models, which delays time-to-value.
- **Inaccurate predictions:** It is a known fact of machine learning that more data leads to greater accuracy. It even trumps a better algorithm. Since large data sets cannot be processed with traditional methods, due to memory and computational limitations, only a subset of the data is often used to train models, which reduces the accuracy of subsequent insights and puts at risk any business decisions based on those insights.
- **Delayed deployment:** Owing to the differences in development and production data environments, deploying predictive models into production is often slow and tedious, and sometimes requires rebuilding everything from scratch in a different technology. This jeopardizes the success of large-scale analytics initiatives and drastically increases time to production.
- **Increased costs:** Additional hardware, software tools, and administrator and developer resources are required for moving data, building duplicate predictive models, and running them on multiple platforms to obtain the desired results.

Built from the ground up to handle massive volumes of data, Vertica addresses the challenges of big data analytics using a balanced, distributed, compressed, columnar paradigm.

In the same way that a bank without databases can't compete with a bank that has them, a company without machine learning can't keep up with ones that use it.

Vertica Analytics Platform Delivers Predictive Analytics at Speed and Scale

Built from the ground up to handle massive volumes of data, Vertica addresses the challenges of big data analytics using a balanced, distributed, compressed, columnar paradigm, Vertica includes many of the most commonly used machine learning algorithms and data preparation, exploration, and model evaluation functions already optimized for distributed systems, and manages models as first-class citizens. The Vertica Analytics Platform eliminates or minimizes some of the barriers to applying ML at scale.

Massively parallel processing enables data to be handled at petabyte or even exabyte scale for your most demanding use cases. Columnar storage capabilities provide data compression and reduce analytics query times down from hours to minutes or minutes to seconds, compared to legacy technologies. In addition, as a full-featured analytics system, Vertica provides advanced SQL-based analytics including pattern matching, geospatial analytics, time-series, and many more capabilities, all built-in.

As an optimized platform enabling advanced predictive modeling to be run from within the database and across large data sets, Vertica eliminates the need for data duplication and processing on alternative platforms—typically requiring multi-vendor offerings—that add complexity and cost. That same speed, scale, and performance used for Business Intelligence (BI) analytics can be applied to machine learning-based predictive analytics.

"Because Vertica is so scalable, we can query an unlimited amount of historical data without the need for pre-aggregation."

YAACOVBEN-YAACOV
Founder and CEO
CatchMedia

Implementing Machine Learning with the Vertica Analytics Platform

Machine learning is most effective when applied to very large data sets and thus is a natural fit for Vertica, which is designed for big data. There are three primary ways of deploying machine learning capabilities in Vertica: Vertica's in-database machine learning algorithms, user-defined extensions (UDxs), and importing external machine learning models through PMML (Predictive Model Markup Language) or TensorFlow's Frozen graph format.

In-Database Machine Learning

Vertica provides in-database machine learning algorithms, including some of the most popular, which can be trained, evaluated and deployed natively on large data sets, accelerating decision making with ease. Built into Vertica's core—with no need to download and install separate packages—the in-database machine learning algorithms deliver:

- **Scalability:** While most external tools like R and Python have limitations on the size of the data set they can handle—forcing users to down-sample—Vertica's in-database machine learning takes advantage of the larger data sets supported to provide more accurate insights.
- **Simplicity:** Vertica's native high speed parallel data ingest, data preparation, and model management capabilities cover the entire data mining lifecycle, eliminating the need to export and load data into another tool for exploration or preparation. In addition, users can train, test, and deploy machine learning models using a familiar SQL interface, or Jupyter and Python. This speeds productivity, reduces time-to-value, and expands the potential user base.
- **Speed:** Vertica's in-database machine learning accelerates model training and prediction by distributing the workload across Vertica's massively parallel processing (MPP) architecture, including the use of multiple nodes in the cluster, and dividing the workload for faster computation when required.

Vertica provides several machine learning functions for performing in-database predictive analytics on large data sets to increase accuracy of predictions and accelerate access to hidden insights.

In-Database Machine Learning Features

Vertica provides many built-in machine learning functions for performing in database predictive analytics on large data sets. Some of the major use cases for machine learning applications revolve around classification, clustering, and prediction. Vertica's built-in machine learning algorithms cover all of these areas with k-means, bisecting k-means, linear regression, logistic regression, support vector machines, random forest, XGBoost, isolation forest, and naïve Bayes, among others.

End-to-End Machine Learning Process in One Place

From data preparation to model scoring and deployment, Vertica supports the entire machine learning process. Users can prepare data with functions for normalization, outlier detection, finding correlations, sampling, imbalanced data processing, missing value imputation, and more. Fast joins of disparate data sets are a normal part of database operations, which greatly shortens data preparation time.

Machine learning models can be trained and tested on massive data sets, and then evaluated with model-level statistics such as ROC curves and confusion matrices. All of this is done with a simple SQL interface that doesn't require specialized programming knowledge, and speeds team productivity.

"We initially chose Vertica because of its superior performance and query speed, but we now realize there is so much more we can do. Vertica's machine learning capabilities will help us analyze network performance, predict capacity constraints, and ensure the best quality of service to our customers."

ALPHONSO LARA
IT Director
Maxcom Telecomunicaciones

"Vertica enables the deep analytics that our product teams use to ensure the best product is being delivered to create the most value for the farmer. Its scalable machine learning is critical in building a data driven culture."

ERICH HOCHMUTH
Big Data Director
The Climate Corporation

Vertica Pyisan opensource library that allows you to use a Jupyter notebook and Python style code to access the in-database Vertica functionality that is similar to Pandas and SciKit Learn. Conduct exploratory and interactive data science on whole data sets stored in Vertica or accessible to Vertica's distributed query engine.

Machine Learning Model Management

Because Vertica trains machine learning models inside the database, it creates a data science repository for all activities that are shared across all data scientists using the platform. With traditional approaches, a model might exist on someone's laptop. If that individual is unavailable, the model maybe inaccessible. Furthermore, without a model repository it can be difficult to compare models, to see how they were generated, and what data sets were used.

In Vertica, models are objects in the database, just like tables. They are visible and readable by anyone with authorized access. And the final feature set that trained the model can be stored with it, in the same database. With thousands of columns permitted, Vertica works well as a feature store allowing feature reuse by other teams, reproducibility, as well as high security and clear provenance.

Models can also be easily exported from a development, test, or sandbox environment into a production environment (or many production environments). Most traditional approaches involve using different technology stacks for development and production. That makes putting models into production very complicated. An in-database machine learning development platform means much of that hassle goes away.

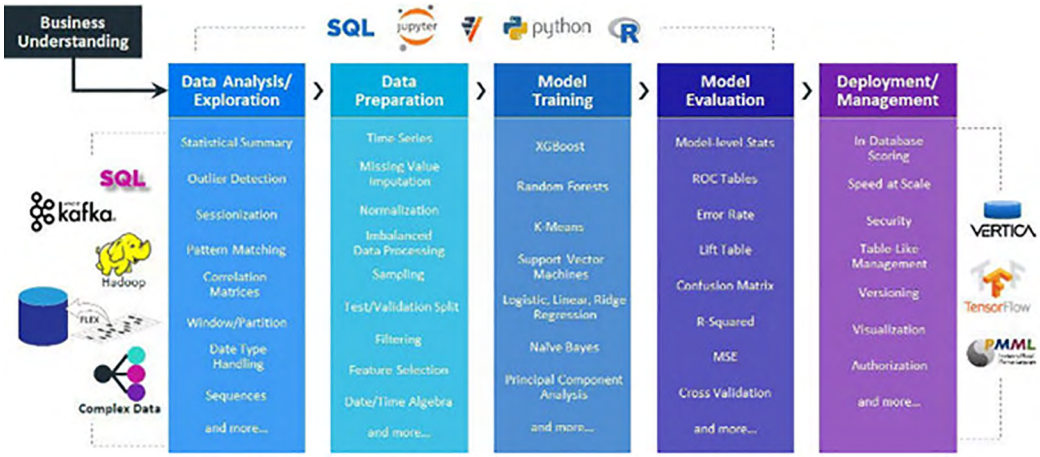

Figure 1. End-to-end machine learning management. From data prep to deployment, Vertica supports the entire machine learning process.

Integration with External Machine Learning

Vertica supports the entire machine learning process. However, in today's business world, multiple tools and programming languages are often used within the same organization for different machine learning purposes. Vertica allows you to work with these tools while also leveraging powerful capabilities, such as model management and distributed execution on very large data sets.

User-Defined Extensions (UDxs)

Vertica connects to hundreds of applications, data sources, ETL tools, and visualization tools—and what it doesn't connect to out of the box can be easily integrated. A UDx allows you to develop your own tools for the Vertica Analytics Platform, including new types of data analysis and the ability to parse and load new types of data. UDxs can be developed in C++, Java, Python, or R using Vertica's powerful SDK.

The broad array of user-defined capabilities (functions, transforms, aggregates, analytics, and loading) leverages the MPP capabilities of Vertica, increasing the power and flexibility of procedural code by bringing it closer to the data (structured, semi-structured, or unstructured). Vertica's user interface makes it easy to deploy and use procedural extensions, which simplifies operational practices and promotes code reuse.

Predictive Model Markup Language (PMML) Support

With Vertica's PMML model import capability, users can build a machine learning model in the tools of their choice and then bring that model to Vertica for management, archiving, and prediction on very large data sets. As a distributed analytical database, Vertica is often home to the most recent (hot) data, or real time data streaming in. Bringing models close to where that data resides provides fast and timely predictions.

Vertica also supports PMML export. With this feature, users can train machine learning models on large data sets, which usually results in more accurate models that are less prone to over-fitting. These models built in Vertica can then be exported for prediction using other tools.

One common use case: Train the model in-database, using the maximum amount of data available. After evaluation and scoring, export the model to an edge node or device to make predictions with the lowest possible latency.

TensorFlow Integration

TensorFlow is a popular open-source machine learning platform that can utilize Graphics Processing Units (GPUs) to train highly sophisticated machine learning models such as neural networks efficiently. Vertica provides the capability to import TensorFlow frozen graph models into Vertica. Once trained externally, these models can be brought to Vertica to sit close to new data as it flows in, making fast predictions. This simplifies the process of putting models into production, minimizes extensive data movement, and can result in significant cost savings for users who may not want to keep the expensive GPU-based infrastructure required to periodically train these models running all the time.

Vertica Built-in Algorithm Examples

New algorithms and other machine learning capabilities such as Jaro-Winkler distance for text comparisons or Yule-Walker method for autocorrelation, are added to every new Vertica release. Below are examples of K-Means and regression, the most common in production ML algorithms in Vertica.

"Using Vertica's User-Defined Extensions, we have already built a bespoke lowpass filter to run against our data—that ensures top analytical performance without having to move the data to other systems or tools."

PHIL CHARLES
Technical Manager
Jaguar TCS Racing
Formula E

"Our Vertica platform is instrumental in many areas of our business—creating predictive algorithms, serving up product recommendations, powering insight to our mobile apps, and generating daily reports and ad-hoc queries. It's crucial for enabling us to be more agile with data."

BRUCE YEN
Director–Business Intelligence
GUESS?, Inc.

The k-means algorithms are a type of unsupervised learning algorithm, meaning the input data is unlabeled. The purpose of k-means and bisecting k-means is to partition *n* observations into *k* clusters.

Clustering with K-Means & Bisecting K-Means

Clustering detects natural groupings within the data, meaning that items within a cluster have more in common with each other than they do with items outside of the cluster. Each of these algorithms takes the unlabeled data and clusters the data points into K different clusters based on similarities between the data points. K-means assigns each observation to the cluster with the nearest mean. That nearest mean is also known as the cluster center. The resulting model can be used to add new data to existing clusters later by assigning it to the cluster with the nearest mean or cluster center.

Bisecting K-means works by dividing the data in half, and then dividing the largest next cluster in half until the data is divided into K clusters. Bisecting K-means works differently but produces similar results. It has the advantage of storing all values of K between 1 and the given K such that each cluster groupings can be easily compared without having to generate new models for each value of K. K-means and bisecting K-means have a wide number of applications, including:

- **Customer segmentation:** Segment customers and buyers into distinct groups (clusters) based on similar attributes such as age, income, or product preferences, to target promotions, provide support, and explore cross-sell opportunities.
- **Fraud detection:** Identify individual observations that don't align to a distinct group (cluster) and identify types of clusters that are more likely to be at risk of fraudulent behavior.

You can apply these algorithms using basic SQL syntax:

```
KMEANS ( 'model_name', 'input_table',
'input_columns', 'num_clusters')
```

Or

```
BISECTING_KMEANS ( 'model_name', 'input_table',
'input_ columns', 'num_clusters')
```

The optional parameters for these functions include an epsilon value (to adjust convergence), maximum number of iterations, three options for initial cluster centers, and an output view. Bisecting k-means also includes optional parameters to control the method and parameters for which bisecting occurs, such as the bisecting method, minimum divisible cluster size, and bisecting iterations.

Function Name	Functionality
kmeans	Find k cluster centers for an input table/view
bisecting_kmeans	Find k cluster centers for an input table/view by bisecting clusters into iterative pairs
get_model_summary	Display the cluster centers along with some evaluation metrics
apply_kmeans	Given an existing k-means model, assign rows from an input table to the defined cluster

Logistic Regression

Logistic regression is used to classify data into two groups based on the logical relationship between independent variables, or features, and some dependent variable, or outcome. The outcome of logistic regression is a binary value which represents an outcome such as true/false, pass/fail, yes/no, or 1/0. Logistic Regression can also be a probability between the two groups. You can use logistic regression models to:

- **Risk analysis:** Use a loan applicant's credit history, income, and loan conditions (predictors) to determine the probability that the applicant will default on a loan (response). The result can be used for approving, denying, or changing loan terms.
- **Predictive maintenance:** The likelihood that a particular mechanical part of a system will malfunction or require maintenance (response) based on operating conditions and diagnostic measurements (predictors).
- **Treatment response prediction:** Determine the likelihood of a patient's successful response to a particular medicine or treatment (response) based on factors like age, blood pressure, smoking, and drinking habits (predictors).

You can apply the logistic regression algorithm using basic SQL syntax:

```
LOGISTIC_REG ( 'model_name', 'input_table',
'response_ column', 'predictor_columns')
```

The optional parameters for this function include optimizer type, regularization method (lasso, ridge or ENet), epsilon value, the lambda and alpha values (for regularization) and the maximum number of iterations. When the model is called (with predict_logistic_reg), you can show either binary output, or a probabilistic output.

Function Name	Functionality
logistic_reg	Build a logistic regression model with the given inputs
get_model_summary	Display the coefficient, intercept, p-value and z-value for the model
predict_logistic_reg	Using the trained logistic regression model, apply the results to a given dataset

Linear Regression

Linear regression is primarily used to predict continuous numerical outcomes in linear relationships along a continuum. Using linear regression, you can model the linear relationship between independent variables, or features, and a dependent variable, or outcome. For example, linear regression models allow you to do the following:

- **Price prediction:** Model residential home prices (response) as a function of the homes' features (predictors), such as living area, number of bedrooms, or number of bath- rooms, etc.
- **Demand prediction:** Model the demand for a service or good (response) based on its features (predictors)—for example, demand for different models of laptops based on monitor size, weight, price, operating system, etc.

Linear regression is primarily used to predict continuous numerical outcomes in linear relationships along a continuum.

These are just examples. Vertica also provides functions for support vector machines (SVM), random forest, XGBoost, SVD, isolation forest, PCA, and others. Learn more at: www.vertica.com/machinelearning

- **Strength/durability prediction:** Determine linear relationship between the compressive strength of concrete (response) and varying amounts of its components (predictors) like cement, slag, fly ash, water, super plasticizer, coarse aggregate, etc.

NaïveBayes

The NaïveBayes algorithm is used to classify data. The algorithm uses different features as predictors to calculate the probability of a specific class, or more than one class. For instance, to predict the probability that an email is spam, you would use words normally associated with spam. The same would be true if you want to classify a document based on content—for example, news, finance, sports, etc.

This supervised machine learning algorithm has several applications, including:

- **Spam detection:** Classifying documents into categories such as email into spam, clutter, social, etc.
- **Text classification:** Classifying news articles into categories likes sports or politics Facial recognition applications.

You can apply the Naïve Bayes algorithm using basic SQL syntax:

```
NAIVE_BAYES ( 'model_name', 'input_table',
'response_ column', 'predictor_columns')
```

Naïve Bayes takes an optional parameter called alpha which can be used to control Laplace smoothing.

Data Preparation for Machine Learning

Data preparation is an important pre-processing step for data analysis, often occupying most of the time to complete an advanced data analysis project. Vertica has a rich set of built-in analytic functions, including interpolation, balancing, pattern matching, event series joins, interpolation, correlation detection, advanced aggregation, outlier detection, and principal component analysis (PCA). These functions support the data preparation process, increase the efficiency of data analysis work, and ultimately speed up time-to-value. This allows data scientists to perform the entire data science workflow right inside the database.

Vertica offers a rich set of data preparation functions, including, but not limited to:

Function Name	Functionality
Balance	Creates an equal distribution of data, either by over, or under-sampling classes
Detect_outliers	Identifies outliers within a given dataset
Impute	Fills in missing values with a mean or mode, based on observed values
Normalize	Normalizes disparate data into a consistent range
Normalize_fit	Creates a normalized model for re-use
One_hot_encoder_fit	Applies one-hot encoding to categorical columns by replacing them with binary vectors
PCA	Principal Component Analysis—a method of feature reduction
SVD	Singular Value Decomposition—another feature reduction algorithm

Model Evaluation

Once models are built, they need to be evaluated. Vertica includes a variety of popular model evaluation functions built-in to allow you to affirm the validity of your regression and classification models. Some evaluation functions are used to evaluate regression models, and others are used for classification or random forest specific evaluations. Many are available. Check the documentation for the available list in latest Vertica version.

Model	Confusuion_ Matrix	Error_Rate	Lift_Table	PRC	ROC	Rsquared	MSE
Logistic Regression	Yes	Yes	Yes	Yes	Yes	N/A	N/A
Naïve Bayes	Yes	Yes	Yes	Yes	Yes	N/A	N/A
RF Classification	Yes	Yes	Yes	Yes	Yes	N/A	N/A
SVM Classification	Yes	Yes	Yes	Yes	Yes	N/A	N/A
Linear Regression	N/A	N/A	N/A	N/A	N/A	Yes	Yes
RF Regression	N/A	N/A	N/A	N/A	N/A	Yes	Yes
SVM Regression	N/A	N/A	N/A	N/A	N/A	Yes	Yes

Cross Validation
Run a trained model against a given dataset through an iteration of "folds," whereby each fold includes a different set of data as training data, and the remaining data as test data. Cross validation then shows the average results against n-folds of the data. Cross validation allows for numerous customizations and for a wide variety of output metrics for evaluation.

What Can You Do with Vertica's In-Database Machine Learning?

The following is an example of a linear regression model that uses the Prestige DataSet.[3] This dataset is used to determine the relationship between income and other variables. In this example, we will demonstrate how to load the dataset and use Vertica's linear reg function to better understand how the predictors impact the response. The example is given in SQL, but the same example can be done in-database using the VerticaPy library.

Some full data science workflow examples using Jupyter, Python, Vertica, and VerticaPy are available here: **www.vertica.com/python/examples/**

The dataset contains the following information:

- Name of occupation Education (in years)
- Income—Average income of incumbents in dollars in 1971
- Women—Percentage of incumbents who are women
- Prestige—The Pineo-Porter prestige score for the occupation, from a social survey conducted in the mid-1960s
- Census—Canadian Census occupational code
- Type—Occupational type, where bc indicates blue collar, wc indicates white collar, and prof indicates professional, managerial, or technical

3 *"PrestigeDataSet." Statistics Canada.Canada (1971) Census of Canada. Vol. 3, Part 6.*

If you need to consider a categorical predictor for training a linear regression model in Vertica, convert it to a numerical value in advance. There are several techniques for converting a categorical variable to a numeric one. For example, you can use the one-hot encoding fit function.

The goal is to build a linear regression model using this data set that predicts income based on the other values in the data set and then evaluate the model's "goodness of fit."

To begin, how do we choose which variables to use for this model? The type column can be eliminated because Vertica does not currently support categorical predictors. Since the occupation and census columns contain many unique values, they are unlikely to predict income under this use case.

That leaves the education, prestige, and women columns.

NOTE: *In practice, if you need to consider a categorical predictor for training a linear regression model in Vertica, convert it to a numerical value in advance. There are several techniques for converting a variable from categorical to numeric. For example, you can use one-hot encoding.*

Step 1. Load the Data

The following shows the table definition to store the Prestige Data Set:

```
=> DROP TABLE IF EXISTS public.prestige CASCADE;
=> CREATE TABLE public.prestige
( occupation VARCHAR(25), education NUMERIC(5,2),
-- avg years of education income INTEGER, -- avg
income
women NUMERIC(5,2), -- % of woman prestige
NUMERIC(5,2),
-- avg prestige rating census INTEGER, --
occupation code
type CHAR(4) -- Professional & managerial
(prof)
) -- White collar (wc)
-- Blue collar (bc)
-- Not Available (na)
```

Vertica supports a number of ways to load data—from files such as delimited text, Parquet, ORCJSON, or from Avro formats, to name a few. To load the data from the Prestige Data Set into the Vertica table, use the following SQL statement:

```
=> COPY public.prestige
FROM
'/home/dbadmin/prestige.txt'
DELIMITER ','
SKIP 1
ABORT ON ERROR DIRECT ;
```

Step 2. Create a Linear Regression Model

Now let's use the Vertica machine learning function LINEAR_REG to create the linear regression model.

To create the model, run the LINEAR_REG function against the public. prestige table as shown. In this statement, income is the response, and the predictors are education, women, and prestige:

```
SELECT LINEAR_REG( 'lnr_prestige',
'public.prestige', 'income', 'education, women,
prestige');
```

This statement is trying to identify the coefficients of the following equation:

$$income = \alpha + \beta_1 education + \beta_2 women + \beta_3 prestige$$

After you create the model, use the GET_MODEL_SUMMARY function to observe the model's properties:

```
SELECT GET_MODEL_SUMMARY(using parameters
model_name = 'lnr_prestige');
```

`GET_MODEL_SUMMARY` returns the following information:

```
-----------+-------------+--------------+----------+---------
predictor |coefficient | std_err    |t_value   |p_value
intercept |-253.84973 |1086.15687 |-0.23371 | 0.81569
education | 177.19903 | 187.63223 | 0.94440 | 0.34729
women     | -50.89570 | 8.55618   |-5.94841 | 0.00000
prestige  | 141.43535 | 29.90961  | 4.72876 | 0.00001
```

Using these coefficients, you can rewrite the equation to read:

```
income = -253.8390442
+177.1907572 * education
-50.950663456 * women
+141.463157 * prestige
```

Finally, let's explore how to measure how well the linear regression model fits the data, or goodness of fit. In Vertica, the `PREDICT_LINEAR_REG` function applies a linear regression model on the input table. You can read more about this function in the Vertica documentation.

Step 3. Evaluate Goodness of Fit

A common method used to test how well your linear regression model fits the observed data is the coefficient of determination. The coefficient is defined in the following equation:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

The coefficient of determination R2 ranges between 0 (no fit) and 1 (perfect fit). To calculate the coefficient of determination, use the Vertica RSQUARED function:

```
SELECT RSQUARED(income, predicted) OVER()
FROM ( SELECT income, PREtDICT_LINEAR_REG
(prestige, women USING PARAMETERS
MODEL_NAME='lnr_prestige') AS predicted

FROM public.prestige

) x ;
```

rsq	Comment
0.63995924449805	Of 102 rows, 102 were used

The evaluation of the coefficient of determination often depends on what area you are investigating. In the social sciences, a coefficient of 0.6 is considered quite good.

When evaluating a model, it is important to consider multiple metrics. A single metric could give you a good value, but the model itself may not be as useful as you need. It is important to understand the Rsquare value, as well as the other metrics, to evaluate the goodness of fit. Vertica provides several model evaluation functions using different metrics.

Use Cases for Machine Learning across Different Industries

The in-database machine learning capabilities of the Vertica Analytics Platform can be used to drive tangible business benefits across abroad range of industries:

- **Financial services** organizations can discover fraud, detect investment opportunities, identify clients with high-risk profiles, or determine the probability of an applicant's defaulting on a loan.
- **Government agencies**, such as public safety and utilities, can use machine learning to help detect fraud, minimize identity theft, and analyze data from smart meters to identify ways to increase efficiency and save money.
- **Communication service providers** can leverage a variety of network probe and sensor data to analyze network performance, predict capacity constraints, and ensure quality service delivery to end customers.
- **Marketing and sales** organizations can use machine learning to analyze buying patterns, segment customers, personalize the shopping experience, and implement targeted marketing campaigns.

- Oil and gas organizations can leverage machine learning to analyze minerals to find new energy sources, streamline oil distribution for increased efficiency and cost-effectiveness, or predict mechanical or sensor failures for proactive maintenance.
- Transportation organizations can analyze trends and identify patterns that can be used to enhance customer service, optimize routes, and increase profitability.

Summary

Vertica's in-database machine learning capabilities allow users to take advantage of big data while simplifying and speeding up their predictive analytics processes to make better-informed decisions, compete more effectively, and accelerate time-to-insight. At the same time, Vertica offers the freedom to create unique capabilities through its support for user defined extensions programmed in C++, Java, Python, or R. Vertica
also supports interactive data science work via VerticaPy, and supports applications that use machine learning via robust APIs for Go, Python, Node.js, and a variety of other languages.

Resources

Try the **Vertica Community Edition** at **vertica.com/try** and get started with your in-data base machine learning initiative. This free edition of Vertica allows you to analyze up to 1 TB of data across 3 nodes on- premises, in the clouds, or in hybrid environments for as long as you like.

Machine Learning for Predictive Analytics, Vertica documentation.

Get free online on-demand video-based training at **Vertica Academy**.

Learn more at
vertica.com/machinelearning
www.opentext.com

Connect with Us

Vertica's in-database machine learning capabilities allow users to take advantage of big data, while simplifying and speeding up their predictive analytics processes to make better-informed decisions, compete more effectively, and accelerate time-to insight.

opentext™