Forward-looking statements
This is a rolling (up to three year) roadmap and is subject to change without notice.

This document contains forward looking statements regarding future operations, product development, product capabilities and availability dates. This information is subject to substantial uncertainties and is subject to change at any time without prior notification. Statements contained in this document concerning these matters only reflect Hewlett Packard Enterprise's predictions and / or expectations as of the date of this document and actual results and future plans of Hewlett Packard Enterprise may differ significantly as a result of, among other things, changes in product strategy resulting from technological, internal corporate, market and other changes. This is not a commitment to deliver any material, code or functionality and should not be relied upon in making purchasing decisions.
Please give me your feedback

**Session ID: B9148**  
**Speaker: Arash Fard and Vincent Xu**

–Use the mobile app to complete a session survey

1. Access “My schedule”
2. Click on the session detail page
3. Scroll down to “Rate & review”

– If the session is not on your schedule, just find it via the Discover app’s “Session Schedule” menu, click on this session, and scroll down to “Rate & Review”

– If you have not downloaded our event app, please go to your phone’s app store and search on “Discover 2016 Las Vegas”

– Thank you for providing your feedback, which helps us enhance content for future events.
Performing Highly Scalable Predictive Analytics with Vertica

11am-11:50am, Wednesday, August 31, 2016

Software Development Manager
Agenda

– Why In-DB Machine Learning
– Machine Learning Functions
  – Data Preparation
  – Linear Regression
  – Logistic Regression
  – K-means Clustering
  – Model Management
– Sales Forecasting using Vertica Machine Learning
– Q&A
Why In-DB Machine Learning
Current Barriers to Machine Learning and Predictive Analytics

- **Added Cost**
  - Additional hardware required for building predictive models

- **Requires Down Sampling**
  - Cannot process large data sets due to memory and computational limitations, resulting in inaccurate predictions

- **Slower Time to Development**
  - Higher turnaround times for model building/scoring and need for moving large volumes of data between systems

- **Slower Time to Deployment**
  - Inability to quickly deploy predictive models into production

#SeizeTheData
Building Machine Learning into the Core of Vertica
New capabilities deliver predictive analytics at speed and scale

- Run in parallel across hundreds of nodes in a Vertica cluster
- Eliminating all data duplication typically required of alternative vendor offerings
- No need to “down-sampling” which can lead to less accurate predictions
- A single system for SQL analytics and Machine Learning
## Machine Learning Pack – 8.0

-- Now installed with Vertica by default

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Model Training</th>
<th>Prediction</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Regression</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>K-means</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model Management</th>
<th>Summarize models</th>
<th>Rename models</th>
<th>Delete models</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Preparation</th>
<th>Normalization</th>
<th>Imbalanced data processing</th>
<th>Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
Data Preparation
Data Preparation – Overview

Data preparation is an important preprocessing step for data analysis, which not only makes it easier to apply machine learning, but also improves the performance and speed of machine learning. Data preparation usually occupies about 60%~70% workload of the whole analysis.

Vertica has a rich set of built-in analytic functions including average, sum, mean, standard deviation, etc. Data scientists can even write their own UDX in R, Python, Java or C++ to prepare the data before training a model.

Vertica will provide a set of functions to support typical data preparation operations, including data normalization, imbalanced data processing, sampling, missing value imputation, outlier detection, feature selection, etc.
Data Preparation – Algorithms

– Normalization
  – Convert the values of different features into similar scale/magnitude, which is essential for algorithms such as k-means, etc.
  – Two methods are provided, MinMax and Z-score.

– Imbalanced Data Processing
  – Balance the number of samples from each class, which is important for training a classification model
  – Weighted-sampling method is supported

– Sampling
  – Select a subset of data to reduce the training data size, which helps reduce the model training time so data scientists can investigate the data set and improve the model faster
  – Random-sampling is supported
In-DB Linear Regression
What is Linear Regression?

- The most basic and commonly used predictive analysis
- To investigate the relationship between one dependent variable (response) and one or more independent variables (predictors)
- To study the strength of the relationship between predictors and the response

\[ y = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3 + \ldots + c_n x_n \]

- \( y \): response
- \( x_i \): predictor
- \( c_i \): coefficient
- \( c_0 \): intercept
## Functions for in-DB Linear Regression Algorithm

<table>
<thead>
<tr>
<th>Function name</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear_reg</td>
<td>Training (2 optimization techniques: BFGS and Newton)</td>
</tr>
<tr>
<td>predict_linear_reg</td>
<td>Scoring</td>
</tr>
<tr>
<td>MSE</td>
<td>Evaluation (mean squared error)</td>
</tr>
<tr>
<td>rsquared</td>
<td>Evaluation (R-squared value)</td>
</tr>
</tbody>
</table>

More detailed information on these functions is available in the Vertica documentation.
Linear Regression Use Cases

**Real Estate**
Model residential home prices \(\text{(response)}\) as a function of the home’s living area, number of bedrooms, number of bathrooms and so on \(\text{(predictors)}\).

**Demand Forecasting**
Model the demand for a service or good \(\text{(response)}\) based on its features \(\text{(predictors)}\); for example, demand for different models of laptops based on monitor size, weight, price, operating system, etc.

**Manufacturing**
Determine linear relationship between the compressive strength of concrete \(\text{(response)}\) and varying amounts of its components \(\text{(predictors)}\) like cement, slag, fly ash, water, super plasticizer, coarse aggregate, etc.
In-DB Logistic Regression
What is Logistic Regression?

- The most basic and popular binary classifier
- A binary class is like: true/false, pass/fail, yes/no, 1/0
- To calculate the likelihood of an outcome (a binary response) based on the values of a set of Independent Variables (predictors)
- To investigate the strength of relationship between predictors and the binary response

\[
\eta = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3 + \ldots + c_n x_n
\]

\[
y = \logit^{-1}(\eta) \quad \text{where} \quad \logit^{-1}(\alpha) = \frac{1}{1+e^{-\alpha}}
\]

y: response
xi: predictor
ci: coefficient
c0: intercept
Functions for in-DB Logistic Regression Algorithm

<table>
<thead>
<tr>
<th>Function name</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>logistic_reg</td>
<td>Training (2 optimization techniques: BFGS and Newton)</td>
</tr>
<tr>
<td>predict_logistic_reg</td>
<td>Scoring</td>
</tr>
<tr>
<td>confusion_matrix</td>
<td>Evaluation</td>
</tr>
<tr>
<td>error_rate</td>
<td>Evaluation</td>
</tr>
<tr>
<td>lift_table</td>
<td>Evaluation</td>
</tr>
<tr>
<td>ROC</td>
<td>Evaluation</td>
</tr>
</tbody>
</table>

More detailed information on these functions is available in the Vertica documentation
Logistic Regression Use Cases

Finance
Use a loan-applicant’s credit history, income, and loan conditions (predictors) to determine probability that applicant will default on loan (response). The result can be used for approving, denying, or changing loans terms.

Engineering
Predicting the likelihood that a particular mechanical part of a system will malfunction or require maintenance (response) based on operating conditions and diagnostic measurements (predictors).

Medicine
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).
K-means Clustering
What is K-means?

– A popular method for cluster analysis

– Partitions $n$ observations into $k$ clusters in which each observation belongs to a specific cluster based on similar properties

– **Grouping 500 items into 5 clusters (k=5)**

Result from Vertica

Result from R k-means
Functions for in-DB K-means Algorithm

<table>
<thead>
<tr>
<th>Function name</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>kmeans</td>
<td>Find k cluster centers for an input table/view, optionally output each row with its assigned cluster</td>
</tr>
<tr>
<td>apply_kmeans</td>
<td>Given an existing k-means model, assign rows from an input table to the correct cluster</td>
</tr>
</tbody>
</table>

More detailed information on these functions is available in the Vertica documentation.
K-means Clustering Use Cases

**Customer Segmentation**
Segment customers and buyers into distinct groups (cluster) based on similar attributes like age, income, product preferences, etc. in order 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.
Model Management
Model Management

List Models

```
SELECT * FROM models;
```

<table>
<thead>
<tr>
<th>schema_name</th>
<th>model_owner</th>
<th>category</th>
<th>model_name</th>
<th>model_type</th>
<th>format_version</th>
<th>model_size</th>
<th>deploy_time</th>
</tr>
</thead>
<tbody>
<tr>
<td>public</td>
<td>dbadmin</td>
<td>Vertica_Models</td>
<td>linearRegModel</td>
<td>linear regression</td>
<td>Vertica Analytic Database v8.0.0-20160622</td>
<td>399</td>
<td>2016-06-22 14:43:01.77-04</td>
</tr>
<tr>
<td>public</td>
<td>dbadmin</td>
<td>Vertica_Models</td>
<td>myKmeansModel</td>
<td>kmeans</td>
<td>Vertica Analytic Database v8.0.0-20160622</td>
<td>845</td>
<td>2016-06-28 12:00:09.51-04</td>
</tr>
</tbody>
</table>

(3 rows)

```
SELECT rename_model('linearRegModel', 'mylinearModel');
```

```
SELECT delete_model('mylinearModel');
```

Summarize Models

```
SELECT summarize_model('LogisticRegModel');
```

Rename Models

```
SELECT rename_model('linearRegModel', 'mylinearModel');
```

Delete Models

```
SELECT delete_model('mylinearModel');
```
Sales Forecasting Using Vertica Machine Learning
Use Case
– Predicting whether a sales deal will win or lose
– According to the prediction results, the company can improve the revenue forecast, adjust sales strategies, and relocate resources, etc.

Algorithm
– Logistic Regression

Data Set
– Sales data of HPE Big Data products, e.g., Vertica, Idol, eDiscovery
– Response
  ▪ Sale stage: Won or Lost
– Predictors
  ▪ Age: Days that a deal has been active
  ▪ Close date delta: Difference between predicted close data and actual close date
  ▪ Close data push counter: Number of times predicted close date changed
  ▪ Rep age: Days that sales rep has been working in the company
  ▪ Rep win rate (%): \( N(\text{won}) / (N(\text{won}) + N(\text{lost})) \)
  ▪ Number of transitions: Number of times the deal got requalified
  ▪ Rep has plan for closing: Yes or No
Sales Forecasting – Continue 1

1. Assume you have a CSV file named verticaSales.csv that contains data. Let’s load data to a table in database.

   ```sql
   CREATE TABLE rawData ( "Sales_Stage" VARCHAR, "Opportunity_ID" VARCHAR, Age INT, Close_Date_Push_Counter INT, "Has_Close_Plan" VARCHAR, Rep_Age INT, "Rep_Win_Rate" FLOAT, "numtransitions" FLOAT);
   COPY rawData FROM LOCAL '/path/verticaSales.csv' PARSER FCSVPARSER(header='TRUE');
   ```

2. Selecting the required columns in any desired order, and converting categorical features to numerical.

   ```sql
   CREATE VIEW bakedData AS
   SELECT Age, DECODE(Sales_Stage, 'Won', 1, 0) as sale, Close_Date_Push_Counter, Rep_Age, Rep_Win_Rate, numtransitions, DECODE(Has_Close_Plan, 't', 1, 0) as closeplan, Opportunity_ID
   FROM rawData;
   ```

3. Using “normalize” function to create a view of normalized data.

   ```sql
   SELECT normalize ('normalData', 'bakedData', '*', 'ZScore',
   --exclude_columns "Opportunity_ID, closeplan, sale"
   --key_columns "Opportunity_ID, closeplan, sale" );
   ```
Sales Forecasting – Continue 2

4. In this example, roughly 75% of the data is used as training data to create a model. The remaining 25% of the data is used as testing data against which you can test your logistic regression model.

```
CREATE TABLE testData AS SELECT * FROM normalData WHERE RANDOM() < 0.25;
CREATE TABLE trainingData AS (SELECT * FROM normalData EXCEPT SELECT * FROM testData);
```

5. Training a logistic regression model.

```
user=> SELECT logistic_reg('logisticRegModel', 'trainingData', 'sale', '*',
'user(> --exclude_columns "Opportunity_ID, sale" ');
logistic_reg
-----------------------------
Finished in 18 iterations

1 row
```

6. Looking at the summary of the trained model.

```
user=> SELECT summarize_model('logisticRegModel');
coeff names: {Intercept, closeplan, age, close_date_push_counter, rep_age, rep_win_rate, numtransitions}
coefficients: {-2.143909192, 0.5457370416, -3.686959884, 1.965739695, 0.1601873911, 0.4097890974, 0.6102197702}
std_err:  {0.15524, 0.27034, 0.22613, 0.15556, 0.09845, 0.091312, 0.10274}
z_value: {-13.81, 2.0187, -16.304, 12.636, 1.6271, 4.4878, 5.9395}
p_value: {2.2166e-43, 0.043515, 9.2051e-60, 1.3288e-36, 0.10372, 7.1969e-06, 2.8583e-09}
Number of iterations: 18, Number of skipped samples: 0, Number of processed samples: 995
Call: logisticReg(model_name=logisticRegModel, input_table=trainingData, response_column=sale, predictor_columns=*, exclude_columns=opportunity_id, sale, optimizer=bfgs, epsilon=0.0001, max_iterations=50)
```
Sales Forecasting – Continue 3

7 Applying the model on the test data.

CREATE TABLE testPrediction AS SELECT Opportunity_ID, sale, predict_logistic_reg(age, close_date_push_counter, rep_age, rep_win_rate, numtransitions, closeplan USING PARAMETERS model_name='logisticRegModel', owner=user', type='response') AS predicted FROM testData;

8 Calculating confusion matrix.

user=> SELECT confusion_matrix(sale::int, predicted::int USING PARAMETERS num_classes=2::int) OVER() FROM testPrediction;

class | 0 | 1 | comment
--------+------+-----+-----------------------------------------------
    0 | 197 |  22 | Of 333 rows, 333 were used and 0 were ignored
    1 |  13 | 101 |
(2 rows)

9 Calculating error rate.

user=> SELECT error_rate(sale::int, predicted::int USING PARAMETERS num_classes=2) OVER() FROM testPrediction;

class | error_rate | comment
--------+-------------+-----------------------------------------------
    0 | 0.100456617772579 |
    1 | 0.114035084843636 |
        | 0.105105102062225 | Of 333 rows, 333 were used and 0 were ignored
(3 rows)
Sales Forecasting – Summary

– Real data, real use case, ~90% accuracy
– Trained, tested and deployed the model within Vertica without having to move data
– Applied a SQL-like interface for ease of use and familiarity
– Don’t forget Vertica UDXes. You can always write your functions in R, Python, Java and C++ to transform data, and evaluate models, etc.
What’s Next?
Continue to Enhance the Predictive Analytics Capabilities

- Support more machine learning functions
  - Naive Bayes
  - Random Forest
  - GBM
  - Neural Network
  - And more…

- Enhance data preparation capabilities
  - Robust z-score normalization
  - Outlier detection
  - Missing value imputation
  - And more…

- Support language bindings for R and Python
  - Data scientists can continue to use their favorite tools, but have Vertica doing the heavy computing behind the scene

This is a rolling (up to 3 year) roadmap and is subject to change without notice
Q&A

vincent.xu@hpe.com
arash.fard@hpe.com
https://my.vertica.com/