Modern programming languages such as Python and R offer developers, data scientists, and analysts a virtually unlimited array of tools, techniques, and algorithms that have allowed businesses to unlock revolutionary value from their data assets.

Unfortunately, leveraging these new capabilities isn’t without compromise. Most practitioners still rely on desktop-scale processing for most tasks. Recent innovations in distributed computing such as Python Dask or Apache Spark have solved some of the scalability challenges but aren’t designed to handle concurrency or complex resource management requirements. Traditional SQL-based Data Warehouse solutions that are designed to scale have been unable to fully support these new data processing requirements, or don’t support these new languages.

Teradata Vantage is the only platform that combines the linear scalability of a world-class Massively Parallel Processing engine, intelligent workload placement, and data locality awareness, with open, flexible support for language of choice including Python and R.

Today’s demonstration uses a publicly available dataset representing energy consumption and weather data from Norway across a three-year period. Or goal will be to prepare a feature set, train a forecasting model, and then load the model into Vantage for high-volume, ultra-low-latency scoring that can be used to power real-time decisioning engines, concurrent usage patterns, or other operational and real-world applications.

In addition, the data prep and feature engineering tasks will also be performed at scale using the Vantage Analytic Library; which contains a large number of powerful whole-data-set statistical, modeling, and analysis functions.

**For the first demonstration**

We’re going to show how a data scientist or developer would access data resident in a remote system. Using the Vantage Python client libraries we create a connection to the target system.

Next, we create a “virtual” dataframe – essentially a pointer to the data resident in the system. This allows us to perform discovery, transformation, and other analysis remotely so we aren’t constrained by the limits of the client system, nor do we have to copy large data sets across the wire.

We’ll see a simple python head method only returns these two rows of data.

**For the second demonstration**

We’re going to show how we can use the Vantage Analytic Library to prepare our analytic data set. As mentioned previously, the Vantage Analytic Library provides an extensive collection of whole-data-set tools; including statistical analysis, hypothesis testing, and algorithmic scoring functions.

For this example, we will use the Transformation functions to prepare the data set. The implementation here is very straightforward in that we create various transformation objects – two one-hot encoders, and a continuous variable scaler. Finally, we define which columns to retain, and then execute the transformation of the data.

The client library pushes this processing down to the Vantage nodes to execute local to the data, and at scale. We’ll see it return in seconds, and then we can check the results.

**For the third demonstration**

We’re going to illustrate how to train the model locally using a common Scikit-learn linear regression pipeline, and then load that trained model into Vantage.

First, we split our test and training sets. Here, we use pythonic methods for splitting the data as it resides in Vantage by slicing the data into the final 24-hour period for testing purposes.

Next, copy the training data to the client for training. To make this process more efficient, the Vantage client libraries will export the data using multiple parallel connections.

Once we have the training data, we use a normal model creation process, and once fit, the model is serialized to PMML format.

Finally, we load the model into Vantage. Note here that the model itself is an entry in the database, stored in it’s own table. This is by design, as it allows enterprises to manage models like any other data set; with information about when it was trained, accuracy scores, versioning, etc. allowing for the operational reporting and management of a multi-model lifecycle.

**For the final demonstration**

We will score and evaluate the model. This process is very straightforward, where we call the Predict function, and pass a pointer to the model we’ve selected, and a pointer to the target data set

(in this case it’s a testing data set, but could be any new data, or data being ingested real-time via Streaming, or even an extremely large dataset representing many countries or regions, or even micro-regions)

Once this function completes, we can analyze the results. This demonstration wasn’t designed for model accuracy, so we can see it’s not that great a result. However, we could also use Vantage Analytic library or other server-side analytical tools to create this sort of analysis at scale.

**Conclusion**

Today we’ve seen how easy it is to use Vantage tools to deploy advanced analytic capabilities directly to data in Vantage; allowing an unlimited breadth of machine learning and analytic techniques to be leveraged at operational scale.