Vertica ML Python Workshop

Setting-up the environment

Ouali Badr

December 2, 2019

Executive Summary

Vertica is one of the fastest advanced analytics database for Big Data and is based on standard SQL ANSI?99. To connect from clients to Vertica we have different options thanks to its standard connection through ODBC, JDBC or ADO.Net. In this post I am going to explain how to configure a connection in MacOS High Sierra to connect a Python client to Vertica through standard ODBC connection, using the Python module PyODBC. As a summary:

- Check Python and ODBC requirements for Vertica
- Download and install ODBC Vertica driver for MacOS
- Configure odbc.ini, odbcinst.ini and vertica.ini files
- Configure environment variables
- Install vertica_python and pyodbc
- Just code in Python!

Contents

"Science knows no country, because knowledge belongs to humanity, and is the torch which illuminates the world."

Louis Pasteur

1 Python client configuration: Windows

1.1 ODBC Driver

Python client connection is supported from Vertica with the standard pyodbc module. Python 2.7.x and 3.x are supported. However as Vertica ML Python works with at least Python 3.6, so you'll need this version installed in your computer.
The ODBC requirements are described in the official Vertica documentation:
`https://my.vertica.com/docs/9.0.x/HTML/index.htm#Authoring/ConnectingToVertica/ClientDriverMisc/ODBCPrerequisites.htm`
Vertica includes its own drivers for different platforms. Just download the drivers from the following url:
`https://my.vertica.com/download/vertica/client-drivers/`
Select your version of Vertica and download the corresponding pkg package to install in your operating system. Vertica client drivers have backwards compatibility since version 8.1.0. So, for example, it is ok if we install version 9 client drivers and connect to 8.1.1 Vertica database. Once the pkg file is dowloaded it can be installed on the system by double clicking in the file.

1.2 Python 3 Installation & Setup Guide

Follow the tutorial at: `https://realpython.com/installing-python/`

1.3 Create a DSN

To create a DSN you just have to do the following steps:

- From the Microsoft Windows Start menu, click Start > Control Panel > Administrative Tools > Data Sources (ODBC).
- Click the System DSN tab and click Add.
- Select the driver that corresponds to your Vertica database and click Finish.

- In the Data source name field, type a data source name.
- Select the same data source name, that you just typed, from the list to register for ODBC. Click OK.
- Select the row containing the data source name that you just created and click Configure.
- Type the database user name and password for your computer.
- Click Connect. The following message is displayed: Connection tested successfully.
- Click OK on all of the open Control Panel windows.

You can now go to the Vertica ML Python installation section.

2 Python client configuration: MacOS / Linux

Even if MacOS is quoted many times in the following sections. The installation of the different tools stay the same for Linux (it is even simpler).

2.1 Python and ODBC requirements

Python client connection is supported from Vertica with the standard pyodbc module for most of the OS platforms and with our own vertica-python module for Linux. Python 2.7.x and 3.x are supported. However as Vertica ML Python works with at least Python 3.6, so you'll need this version installed in your computer.
Check first all the supported versions in Vertica documentation:
`https://my.vertica.com/docs/9.0.x/HTML/index.htm#Authoring/SupportedPlatforms/PerlandPython.htm`
The ODBC requirements are described in the official documentation as well:
`https://my.vertica.com/docs/9.0.x/HTML/index.htm#Authoring/ConnectingToVertica/ClientDriverMisc/ODBCPrerequisites.htm`
Python distribution is embedded in MacOS, but some of the package installations and customization can be painful because Apple doesn't allow you to change or update by default some specific files in the system, just because Python installation included is part of the system. So, my recommendation is to use an external distribution isolated from the Python MacOS system, like Anaconda package, that it can be downloaded from: `https://www.anaconda.com/download/#macos` Another way to go to have a completed isolated python implementation would be using Python Virtualenvs, but we are not going to use it in this example.

So, first is checking Python version with:

```
root@ubuntu:~$ python3 --version

# Output
Python 3.6.3
```

Do not forget to write the command `python3` as `python` will most of the time represent `python2`. Python 2.7.13 version is installed in the system. This is supported for Vertica 9 (the one tested here). Also checking PIP version:

```
root@ubuntu:~$ pip3 --version

# Output
pip 19.3.1 from /Library/Frameworks/Python.framework/Versions/3.6/lib/python3
    .6/site-packages/pip (python 3.6)
```

It is recommended to update PIP version in order to install pyodbc module successfully:

```
root@ubuntu:~$ sudo -H pip3 install --upgrade pip

# Output
Requirement already up-to-date: pip in /Library/Frameworks/Python.framework/
    Versions/3.6/lib/python3.6/site-packages (19.3.1)
```

2.2 Download and install ODBC drivers for Vertica

Vertica includes its own drivers for different platforms, included MacOS. Just download the drivers from the following url:
`https://my.vertica.com/download/vertica/client-drivers/`
Select your version of Vertica and download the corresponding pkg package to install in your operating system. Vertica client drivers have backwards compatibility since version 8.1.0. So, for example, it is ok if we install version 9 client drivers and connect to 8.1.1 Vertica database. Once the pkg file is dowloaded it can be installed on the system by double clicking in the file. Be aware that the package is downloaded from an unknown developer, so MacOS is going to ask for permission to install it. To do this just go to System Preferences > Security and Privacy > General and accept the installation. If we didn't change the location of the installation, the new Vertica drivers should be installed in folder "lib" located at "/Library/Vertica/ODBC/lib". Location can be checked in the Terminal:

```
root@ubuntu:~$ ls /Library/Vertica/ODBC/lib/

# Output
libverticaodbc.dylib  vertica.ini
```

There are two files:

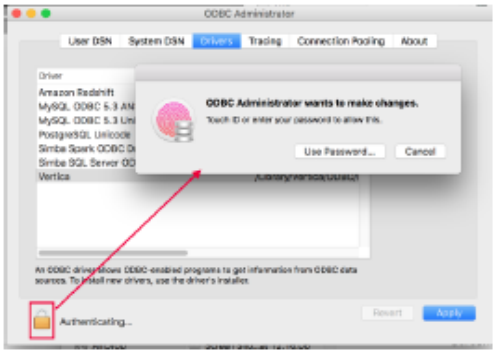
- libverticaodbc.dylib - This is the driver
- vertica.ini - Vertica configuration file template for the driver

2.3 Configure INI files for the ODBC driver

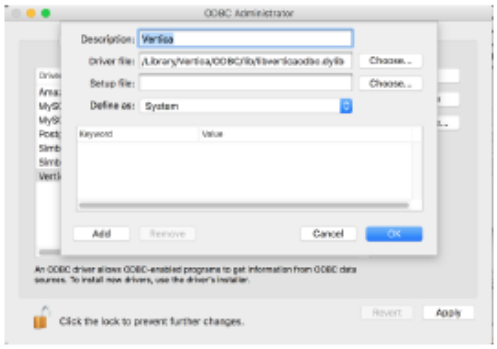
Once Vertica driver is installed, next step is to customize configuration files in order to use customized DSN, encodings and drivers. The following just need to be configured:

- /Library/Vertica/ODBC/lib/vertica.ini
- /Library/ODBC/odbc.ini
- /Library/ODBC/odbcinst.ini
- VERTICAINI and ODBCINI system parameters

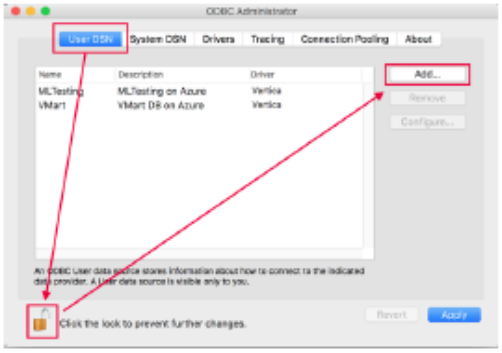
The easiest way to do this is to use ODBC Administrator, a GUI Tool to configure the odbc.ini and odbcinst.ini The ODBC Administrator can be downloaded at: `https://support.apple.com/kb/DL895?locale=en_US`



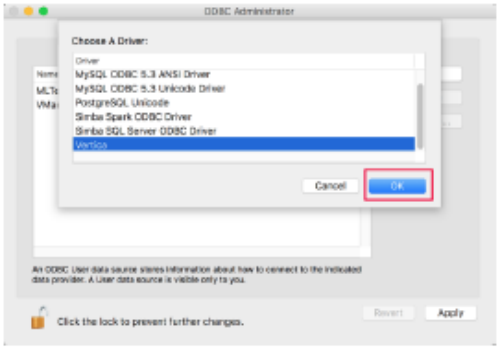
(a) Let's authenticate to make changes to the ODBC file



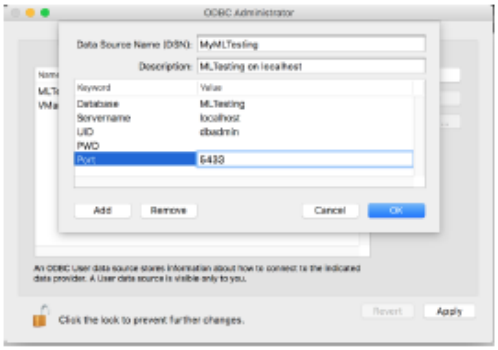
(b) Important that Vertica driver location is loaded from where it was installed (as commented before)



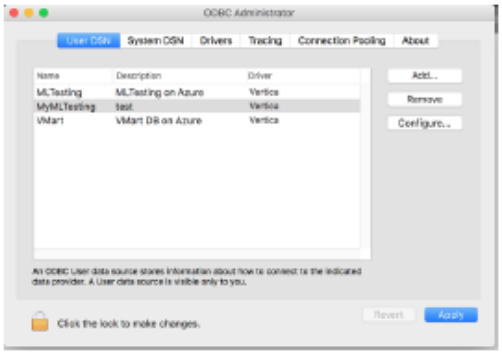
(c) We need to add DSN values with the Database parameters we want to connect



(d) Select the vertica driver for the DSN connection



(e) Add Database, Servername, UID (user), PWD (password) and Port parameters



(f) Check that DSN is configured and Apply

2.4 Odbc.ini and odbcinst.ini configuration

Once downloaded and installed we add a DSN double clicking ODBC Administrator application (in Applications/Utilities/ODBC Administrator) and just as the following:

We check then that odbc.ini has the DSN added previously (MLTesting database example used):

```
root@ubuntu:~$ cat ~/Library/ODBC/odbc.ini

# Output
[ODBC Data Sources]
VMart = Vertica
MLTesting = Vertica
MyMLTesting = Vertica

[ODBC]
Trace = 0
TraceAutoStop = 0
TraceFile =
TraceLibrary =
?.

[MLTesting]
Driver = /Library/Vertica/ODBC/lib/libverticaodbc.dylib
Description = MLTesting on Azure
Servername = vazure
Database = MLTesting
UID = dbadmin
PWD =
Port = 5433
```

Now check that the Vertica driver is in odbcinst.ini:

```
root@ubuntu:~$ cat ~/Library/ODBC/odbcinst.ini

# Output
;# If you did not install in the default directory, replace
'/opt/amazon/redshift'
;# with the correct location.
[ODBC Drivers]
Amazon Redshift = Installed
Vertica = Installed
...
[Vertica]
Driver = /Library/Vertica/ODBC/lib/libverticaodbc.dylib
```

2.5 Vertica.ini configuration

The Vertica ini file is already configured by default, but I recommend to copy to the same place as the other odbc files and change the encoding configuration as follows:

```
root@ubuntu:~$ cp /Library/Vertica/ODBC/lib/vertica.ini ~/Library/ODBC/
# Change the encoding parameter from UTF-32 to UTF-16 in the copied vertica.
    ini:
[Driver]
ErrorMessagesPath=/Library/Vertica/ODBC/messages/
ODBCInstLib=/usr/lib/libiodbcinst.dylib
DriverManagerEncoding=UTF-16
```

2.6 OS Environment variables

Last step to configure odbc to be working with DSNs is to add the environment variable VERTICAINI and ODBCINI. Just add them into your bash_profile (or /etc/profile for system wide):

```
root@ubuntu:~$ vim ~/.bash_profile

## Adding some parameters to work with ODBC for Vertica
[?]
export ODBCINI=/Users/badr/Library/ODBC/odbc.ini
export VERTICAINI=/Users/badr/Library/ODBC/vertica.ini
[?]
```

3 Vertica ML Python API Installation

3.1 Prerequires

3.2 Python Version

`vertica-ml-python` works with at least:

- **Vertica:** 9.1 (with previous versions, some functions and algorithms may not be available)
- **Python:** 3.6

`vertica-ml-python` library is only using the standard Python libraries such as `matplotlib`, `numpy`... Other libraries can be used as `anytree` for tree visualization or `sqlparse` for SQL indentation but they are optional.

3.3 Installation

To install `vertica-ml-python`, you can use the pip command:

```
root@ubuntu:~$ pip3 install vertica_ml_python
```

You can also drag and drop the `vertica_ml_python` folder in the `site-package` folder of the Python framework. In the MAC environment, you can find it in:
`/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-packages`

Another way is to call the library from where it is located.

You can then import each library element using the usual Python syntax.

```
# to import the vDataframe
from vertica_ml_python import vDataframe
# to import the Logistic Regression
from vertica_ml_python.learn.linear_model import LogisticRegression
```

Everything is well detailed in the following documentation.

3.4 Connection to the Database

3.4.1 ODBC

To install `pyodbc`, you can use the pip command:

```
root@ubuntu:~$ pip3 install pyodbc
```

The same for vertica_python:

```
root@ubuntu:~$ pip3 install vertica_python
```

To connect to the database, the user can use an ODBC connection to the Vertica database. `vertica-python` and `pyodbc` provide a cursor that will point to the database. It will be used by the `vertica-ml-python` to create all the different objects.

```
#
# vertica_python
#
import vertica_python

# Connection using all the DSN information
conn_info = {'host': "10.211.55.14", 'port': 5433, 'user': "dbadmin", '
    password': "XxX", 'database': "testdb"}
cur = vertica_python.connect(** conn_info).cursor()

# Connection using directly the DSN
from vertica_ml_python.utilities import to_vertica_python_format # This
    function will parse the odbc.ini file
dsn = "VerticaDSN"
cur = vertica_python.connect(** to_vertica_python_format(dsn)).cursor()

#
# pyodbc
```

```
#
import pyodbc

# Connection using all the DSN information
driver = "/Library/Vertica/ODBC/lib/libverticaodbc.dylib"
server = "10.211.55.14"
database = "testdb"
port = "5433"
uid = "dbadmin"
pwd = "XxX"
dsn = ("DRIVER={}; SERVER={}; DATABASE={}; PORT={}; UID={}; PWD={};").format(
    driver, server, database, port, uid, pwd)
cur = pyodbc.connect(dsn).cursor()

# Connection using directly the DSN
dsn = ("DSN=VerticaDSN")
cur = pyodbc.connect(dsn).cursor()
```

3.4.2 JDBC

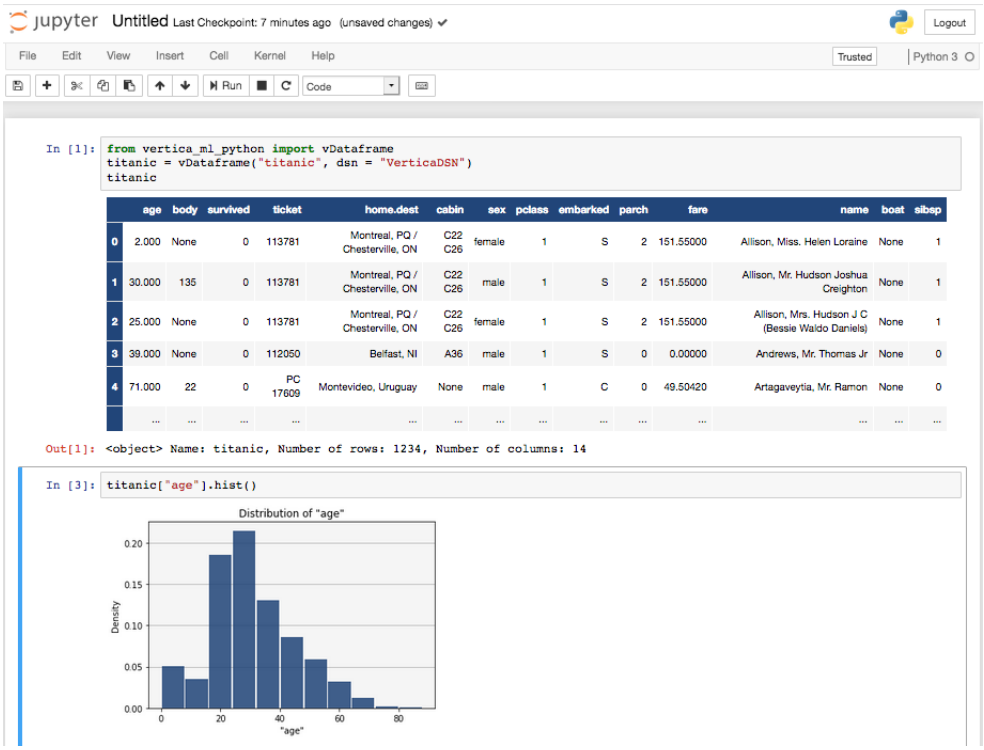
The user can also use a JDBC connection to the Vertica Database.

```
import jaydebeapi

uid = "dbadmin"
pwd = "XxX"
driver = "/Library/Vertica/JDBC/vertica-jdbc-9.0.1-0.jar" #Path to JDBC Driver
url = 'jdbc:vertica://10.211.55.14:5433/'
name = 'com.vertica.jdbc.Driver'
cur = jaydebeapi.connect(name,[url,uid,pwd],driver).cursor()
```

4 Jupyter

Jupyter offers a really beautiful interface to use `vertica-ml-python`.



	age	body	survived	ticket	home.dest	cabin	sex	pclass	embarked	parch	fare	name	boat	sibsp
0	2.000	None	0	113781	Montreal, PQ / Chesterville, ON	C22 C26	female	1	S	2	151.55000	Allison, Miss. Helen Loraine	None	1
1	30.000	135	0	113781	Montreal, PQ / Chesterville, ON	C22 C26	male	1	S	2	151.55000	Allison, Mr. Hudson Joshua Creighton	None	1
2	25.000	None	0	113781	Montreal, PQ / Chesterville, ON	C22 C26	female	1	S	2	151.55000	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	None	1
3	39.000	None	0	112050	Belfast, NI	A36	male	1	S	0	0.00000	Andrews, Mr. Thomas Jr	None	0
4	71.000	22	0	PC 17609	Montevideo, Uruguay	None	male	1	C	0	49.50420	Artagaveytia, Mr. Ramon	None	0
	...	...	...	...	...	...	...	...	...	...	...	...	...	...

All the exercises will be done using Jupyter. You can install and launch the software using the following command:

```
# Installing Jupyter
root@ubuntu:~$ pip3 install jupyterlab
# Lauching Jupyter
root@ubuntu:~$ jupyter notebook
```

The environment is now ready, you can start exercise 0. Happy Playing!