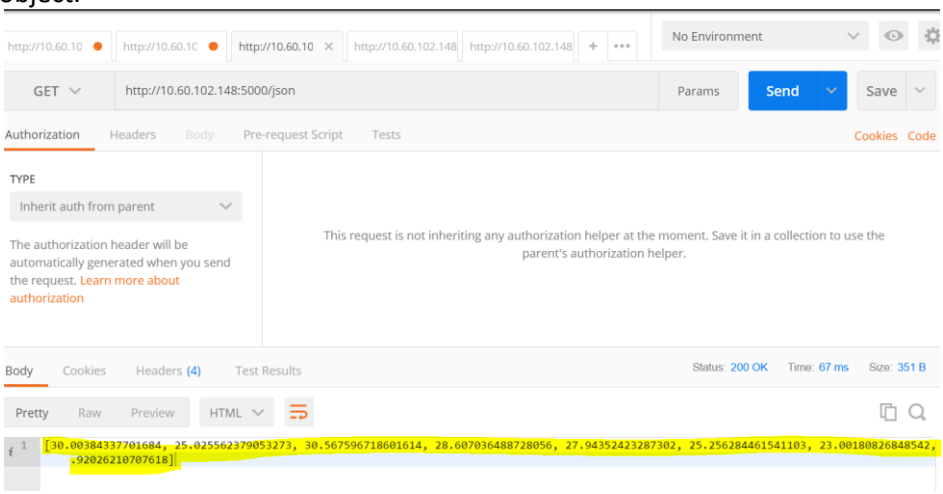
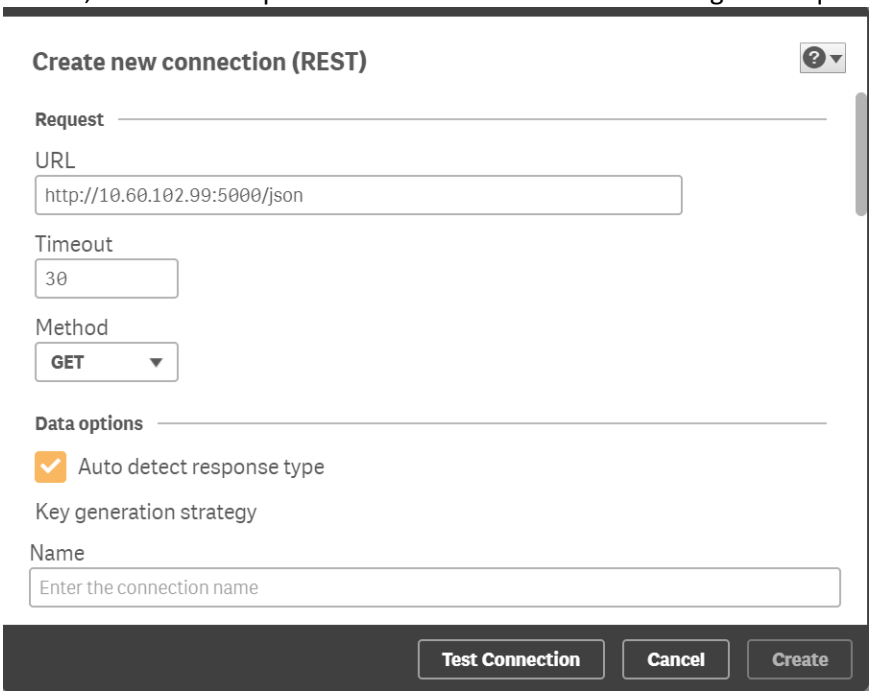
```
ds@ds-VirtualBox: ~/Desktop/regression
ds@ds-VirtualBox:~/Desktop/regression$ atom .
ds@ds-VirtualBox:~/Desktop/regression$ python makeprediction.py
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [13/Apr/2018 10:59:12] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Apr/2018 10:59:13] "GET /favicon.ico HTTP/1.1" 404 -
.. reading file done
MSE: -34.705 (45.574)
[30.00384338 25.02556238 30.56759672 28.60703649 27.94352423 25.25628446
 23.00180827 19.53598843 11.52363685 18.92026211]
127.0.0.1 - - [13/Apr/2018 10:59:17] "GET /json HTTP/1.1" 200 -
```

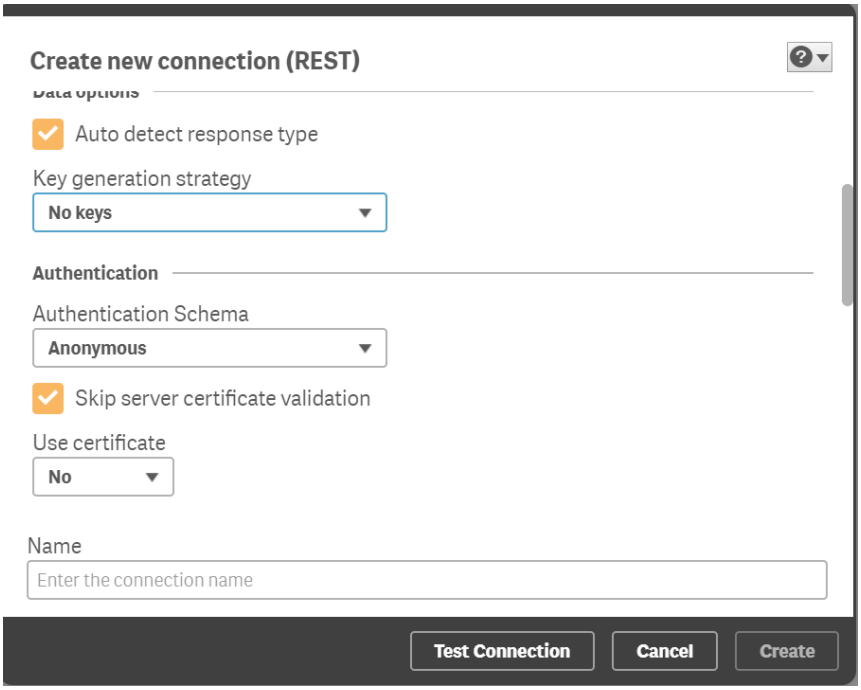
On the server we run the web application that does the predictions and returns the predicted data in the form of a json object.



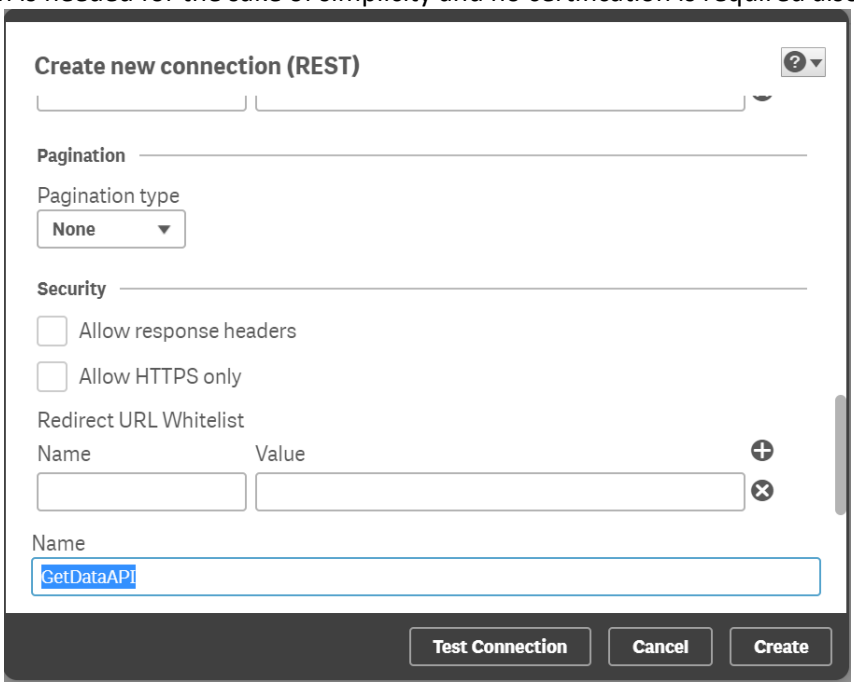
After sending a get request to the local web server pointing the desired endpoint (in this case '/json') we get a list of 8 numeric values, these are the predicted values that we want to integrate in qlik sense.



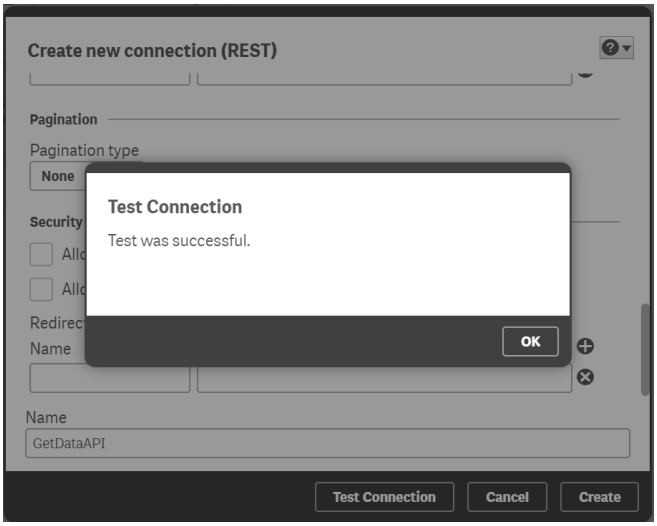
Now, we create a new API connector inside qlik sense to get the predicted data from the web server.



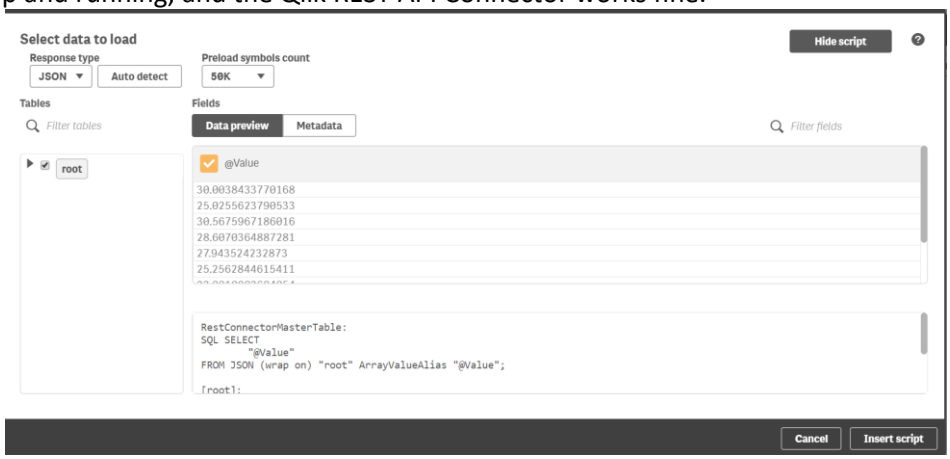
No authentification is needed for the sake of simplicity and no certification is required also.



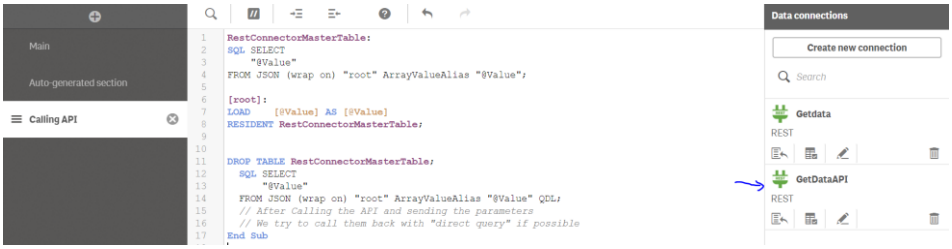
We give our connector a name «GetDataAPI ».



The server is up and running, and the Qlik REST API Connector works fine.



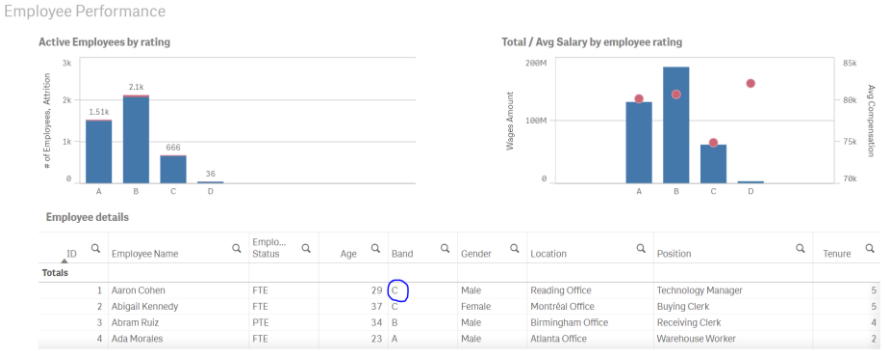
Now, after we create the connector, Qlik sense recognizes the structure of the data and gets them into one dimension.



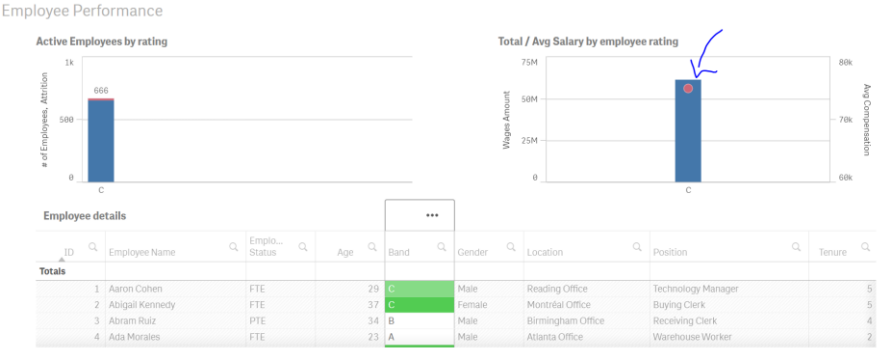
This is the code generated by Qlik Sense

```
Sub CallAPI(x)
// Get selected items from the GUI
// Calling the API Endpoint with the needed arguments
// Getting the new predicted data
// Updating the dimension into the data model
End Sub
```

All these steps, are done manually, we want to create a function or sub routine that automate these steps, and the goal is to get predictions from the server each time the user interacts with the sheet elements, So when a variable is selected the script sends a get request to the server and gets the new predicted data and displays it again on the screen.



For Example, we want when we click on a row on the band column, that triggers a recall to the api and updates the dimension with the new predicted data given that element sent to the server to make the prediction based on it.



In the end we display the new filtered data and charts based on the new dimension gotten from the server after the Click event.