Singular-value decomposition :
========================

https://machinelearningmastery.com/singular-value-decomposition-for-machine-learning/

```
from numpy import array
from scipy.linalg import svd

# define a matrix
A = array([[1, 2], [3, 4], [5, 6]])
print("A : {}".format(A))

# SVD
U, s, V = svd(A)
print("U : {}".format(U))
print("S : {}".format(s))
print("V : {}".format(V))
```

Result :
======

```
A : [[1 2]
     [3 4]
     [5 6]]

U : [[-0.2298477   0.88346102  0.40824829]
     [-0.52474482  0.24078249 -0.81649658]
     [-0.81964194 -0.40189603  0.40824829]]

S : [9.52551809 0.51430058]

V : [[-0.61962948 -0.78489445]
     [-0.78489445  0.61962948]]
```

==

Calculate Singular-Value Decomposition

The SVD can be calculated by calling the svd() function.

The function takes a matrix and returns the U, Sigma and V^T elements. The Sigma diagonal matrix is returned as a vector of singular values. The V matrix is returned in a transposed form, e.g. V.T.

The example below defines a 3×2 matrix and calculates the Singular-value decomposition.

```python
# Singular-value decomposition
from numpy import array
from scipy.linalg import svd
# define a matrix
A = array([[1, 2], [3, 4], [5, 6]])
print(A)
# SVD
U, s, VT = svd(A)
print(U)
print(s)
print(VT)
```

Running the example first prints the defined 3×2 matrix, then the 3×3 U matrix, 2 element Sigma vector, and 2×2 V^T matrix elements calculated from the decomposition.

```
[[1 2]
 [3 4]
 [5 6]]

[[-0.2298477   0.88346102  0.40824829]
 [-0.52474482  0.24078249 -0.81649658]
 [-0.81964194 -0.40189603  0.40824829]]

[ 9.52551809  0.51430058]

[[-0.61962948 -0.78489445]
 [-0.78489445  0.61962948]]
```

==