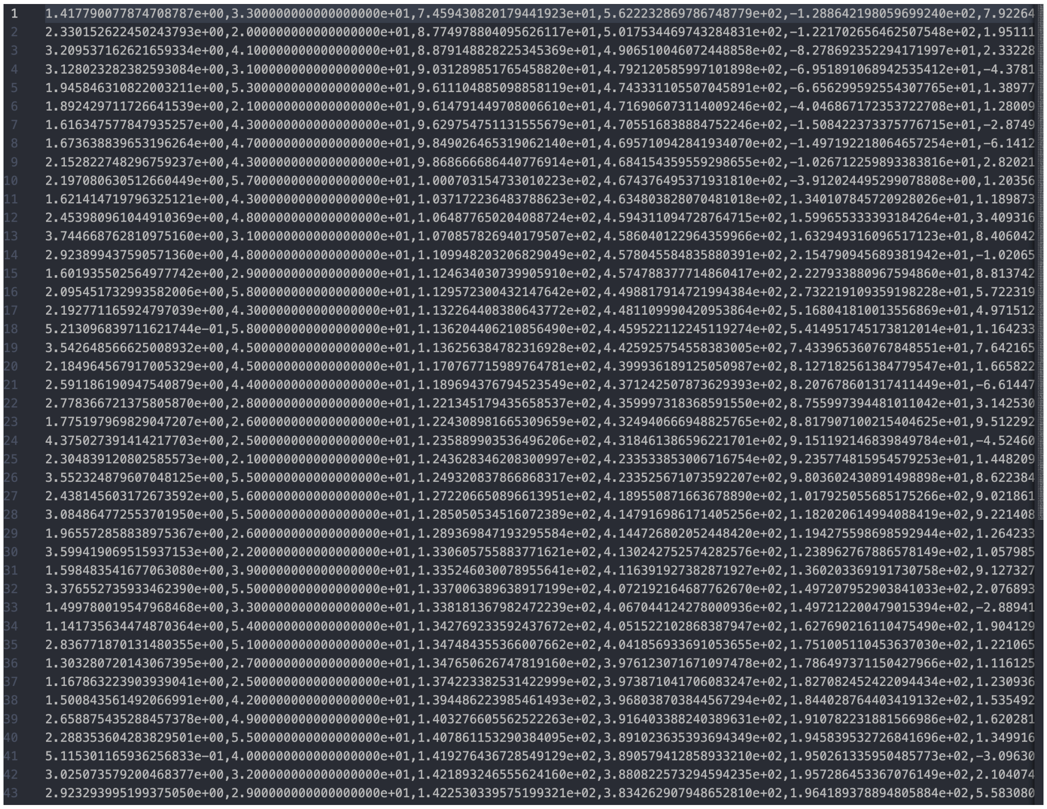
Machine Learning Supervised

Artificial dataset generation

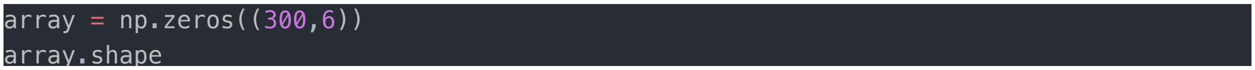


```
1.417790077874708787e+00,3.300000000000000000e+01,7.459430820179441923e+01,5.622232869786748779e+02,-1.288642198059699240e+02,7.92264
2.330152622450243793e+00,2.000000000000000000e+01,8.774978804095626117e+01,5.017534469743284831e+02,-1.221702656462507548e+02,1.95111
3.209537162621659334e+00,4.100000000000000000e+01,8.879148828225345369e+01,4.906510046072448858e+02,-8.278692352294171997e+01,2.33228
3.128023282382593084e+00,3.100000000000000000e+01,9.031289851765458820e+01,4.792120585997101898e+02,-6.951891068942535412e+01,-4.3781
1.945846310822003211e+00,5.300000000000000000e+01,9.611104885098858119e+01,4.743331105507045891e+02,-6.656299592554307765e+01,1.38977
1.892429711726641539e+00,2.100000000000000000e+01,9.614791449708006610e+01,4.716906073114009246e+02,-4.046867172353722708e+01,1.28009
1.616347577847935257e+00,4.300000000000000000e+01,9.629754751131555679e+01,4.705516838884752246e+02,-1.508422373375776715e+01,-2.8749
1.673638839653196264e+00,4.700000000000000000e+01,9.849026465319062140e+01,4.695710942841934070e+02,-1.497192218064657254e+01,-6.1412
2.152822748296759237e+00,4.300000000000000000e+01,9.868666686440776914e+01,4.684154359559298655e+02,-1.026712259893383816e+01,2.82021
2.197080630512660449e+00,5.700000000000000000e+01,1.000703154733010223e+02,4.674376495371931810e+02,-3.912024495299078808e+00,1.20356
1.621414719796325121e+00,4.300000000000000000e+01,1.037172236483788623e+02,4.634803828070481018e+02,1.340107845720928026e+01,1.189873
2.453980961044910369e+00,4.800000000000000000e+01,1.064877650204088724e+02,4.594311094728764715e+02,1.599655333393184264e+01,3.409316
3.744668762810975160e+00,3.100000000000000000e+01,1.070857826940179507e+02,4.586040122964359966e+02,1.632949316096517123e+01,8.406042
2.923899437590571360e+00,4.800000000000000000e+01,1.109948203206829049e+02,4.578045584835880391e+02,2.154790945689381942e+01,-1.02065
1.601935502564977742e+00,2.900000000000000000e+01,1.124634030739905910e+02,4.574788377714860417e+02,2.227933880967594860e+01,8.813742
2.095451732993582006e+00,5.800000000000000000e+01,1.129572300432147642e+02,4.498817914721994384e+02,2.732219109359198228e+01,5.722319
2.192771165924797039e+00,4.300000000000000000e+01,1.132264408380643772e+02,4.481109990420953864e+02,5.168041810013556869e+01,4.971512
5.213096839711621744e-01,5.800000000000000000e+01,1.136204406210856490e+02,4.459522112245119274e+02,5.414951745173812014e+01,1.164233
3.542648566625008932e+00,4.500000000000000000e+01,1.136256384782316928e+02,4.425925754558383005e+02,7.433965360767848551e+01,7.642165
2.184964567917005329e+00,4.500000000000000000e+01,1.170767715989764781e+02,4.399936189125050987e+02,8.127182561384779547e+01,1.665822
2.591186190947540879e+00,4.400000000000000000e+01,1.189694376794523549e+02,4.371242507873629393e+02,8.207678601317411449e+01,-6.61447
2.778366721375805870e+00,2.800000000000000000e+01,1.221345179435658537e+02,4.359997318368591550e+02,8.755997394481011042e+01,3.142530
1.775197969829047207e+00,2.600000000000000000e+01,1.224308981665309659e+02,4.324940666948825765e+02,8.817907100215404625e+01,9.512292
4.375027391414217703e+00,2.500000000000000000e+01,1.235889903536496206e+02,4.318461386596221701e+02,9.151192146839849784e+01,-4.52460
2.304839120802585573e+00,2.100000000000000000e+01,1.243628346208300997e+02,4.233533853006716754e+02,9.235774815954579253e+01,1.448209
3.552324879607048125e+00,5.500000000000000000e+01,1.249320837866868317e+02,4.233525671073592207e+02,9.803602430891498898e+01,8.622384
2.438145603172673592e+00,5.600000000000000000e+01,1.272206650896613951e+02,4.189550871663678890e+02,1.017925055685175266e+02,9.021861
3.084864772553701950e+00,5.500000000000000000e+01,1.285050534516072389e+02,4.147916986171405256e+02,1.182020614994088419e+02,9.221408
1.965572858838975367e+00,2.600000000000000000e+01,1.289369847193295584e+02,4.144726802052448420e+02,1.194275598698592944e+02,1.264233
3.599419069515937153e+00,2.200000000000000000e+01,1.330605755883771621e+02,4.130242752574282576e+02,1.238962767886578149e+02,1.057985
1.598483541677063080e+00,3.900000000000000000e+01,1.335246030078955641e+02,4.116391927382871927e+02,1.360203369191730758e+02,9.127327
3.376552735933462390e+00,5.500000000000000000e+01,1.337006389638917199e+02,4.072192164687762670e+02,1.497207952903841033e+02,2.076893
1.499780019547968468e+00,3.300000000000000000e+01,1.338181367982472239e+02,4.067044124278000936e+02,1.497212200479015394e+02,-2.88941
1.141735634474870364e+00,5.400000000000000000e+01,1.342769233592437672e+02,4.051522102868387947e+02,1.627690216110475490e+02,1.904129
2.836771870131480355e+00,5.100000000000000000e+01,1.347484355366007662e+02,4.041856933691053655e+02,1.751005110453637030e+02,1.221065
1.303280720143067395e+00,2.700000000000000000e+01,1.347650626747819160e+02,3.976123071671097478e+02,1.786497371150427966e+02,1.116125
1.167863223903939041e+00,2.500000000000000000e+01,1.374223382531422999e+02,3.973871041706083247e+02,1.827082452422094434e+02,1.230936
1.500843561492066991e+00,4.200000000000000000e+01,1.394486223985461493e+02,3.968038703844567294e+02,1.844028764403419132e+02,1.535492
2.658875435288457378e+00,4.900000000000000000e+01,1.403276605562522263e+02,3.916403388240389631e+02,1.910782231881566986e+02,1.620281
2.288353604283829501e+00,5.500000000000000000e+01,1.407861153290384095e+02,3.891023635393694349e+02,1.945839532726841696e+02,1.349916
5.115301165936256833e-01,4.000000000000000000e+01,1.419276436728549129e+02,3.890579412858933210e+02,1.950261335950485773e+02,-3.09630
3.025073579200468377e+00,3.200000000000000000e+01,1.421893246555624160e+02,3.880822573294594235e+02,1.957286453367076149e+02,2.104074
2.923293995199375050e+00,2.900000000000000000e+01,1.422530339575199321e+02,3.834262907948652810e+02,1.964189378894805884e+02,5.583080
```

Import necessary modules

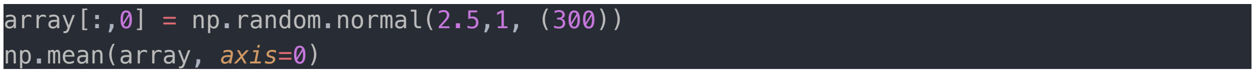
```python
#!/usr/bin/python3

from sys import argv, exit

import numpy as np
```

Create an array containing 0 with 300 datapoints and 6 columns



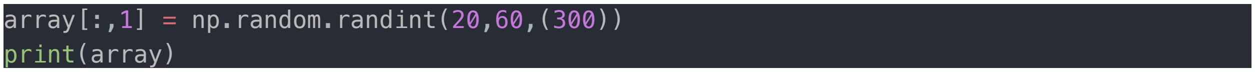
```
array = np.zeros((300,6))
array.shape
```

Change the values in a column to numbers that have a mean of 2.5



```
array[:,0] = np.random.normal(2.5,1, (300))
np.mean(array, axis=0)
```

Change the values of a column by random Integers

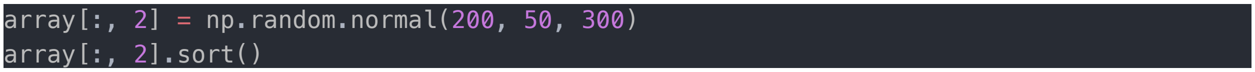


```
array[:,1] = np.random.randint(20,60,(300))
print(array)
```

Result:

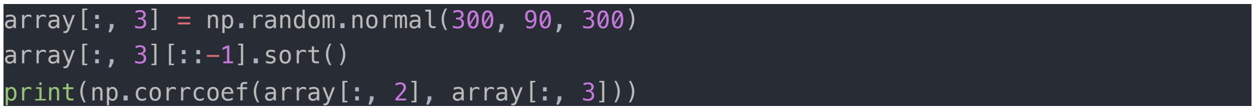
```
[[ 1.41779008 33.        0.        0.        0.        0.      ]
 [ 2.33015262 20.        0.        0.        0.        0.      ]
 [ 3.20953716 41.        0.        0.        0.        0.      ]
 ...
 [ 0.24013131 31.        0.        0.        0.        0.      ]
 [-2.79370555 34.        0.        0.        0.        0.      ]
 [ 1.01028494 32.        0.        0.        0.        0.      ]]
```

Change the values of a column to sorted floats



```
array[:, 2] = np.random.normal(200, 50, 300)
array[:, 2].sort()
```

Create negative correlation

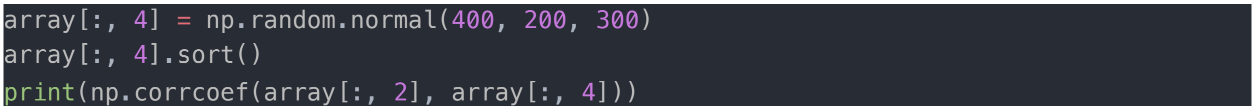


```
array[:, 3] = np.random.normal(300, 90, 300)
array[:, 3][::-1].sort()
print(np.corrcoef(array[:, 2], array[:, 3]))
```

Result:

```
[[ 1.        -0.9930233]

 [-0.9930233  1.       ]]
```

Create positive correlation

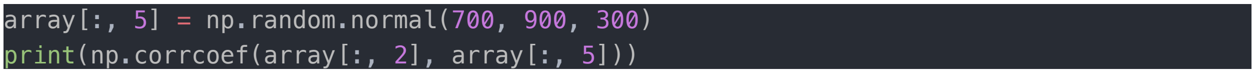


```
array[:, 4] = np.random.normal(400, 200, 300)
array[:, 4].sort()
print(np.corrcoef(array[:, 2], array[:, 4]))
```

Result:

```
[[1.         0.99618073]

 [0.99618073 1.        ]]
```

Create a correlation close to 0



```
array[:, 5] = np.random.normal(700, 900, 300)
print(np.corrcoef(array[:, 2], array[:, 5]))
```

Result:

```
[[1.         0.01673262]

 [0.01673262 1.        ]]
```

Check if the columns all have a different mean

```
np.mean(array, axis=0)
```

Check if the columns all have a different standard deviation

```
array.std(axis=0)
```

Save dataset into a csv file and exit with code 0

```
np.savetxt("artificial_dataset.csv", array, delimiter=",")

exit(0)
```