1. Numpy
   1. Numpy Basic
2. Numpy array creation
3. 1d

a = np.array([1,2,3]);

1. 2d

a2 =np.array([[1,2,3,4],[5,6,7,8]]);

1. All zeros, call function zeros and pass in a tuple

np.zeros((2,2));

1. random

randn(dimension1, dimension2)

np.random.randn(2, 4)

1. Array properties
2. .ndim # number of dimension
3. .shape # length for each dimension

Returns a tuple.

1. .size # total number of elements
2. Index
3. Bracket index,

a2[1,2]

1. Bracket index with slicing

a2[:1] # each row, but 2nd element in col

1. Step slicing

a2[0,3,2] #starting index, stopping index(exclusive), step

1. Step slicing with negative

A2[-1,1,2] # -1 means n-1, in other words, starting from the end.

1. Element wise operations
2. Add: a+2 # add 2 to each element of a
   1. Numpy for Matrix
   2. Matrix multiplication

  mat1 = ([1, 6, 5],[3 ,4, 8],[2, 12, 3])

mat2 = ([3, 4, 6],[5, 6, 7],[6,56, 7])

   res = np.dot(mat1,mat2)

* 1. Matrix transpose, inverse

Numpy.transpose(matrix)

# Inverse of matrix A

print("\nInverse of A:\n", numpy.linalg.inv(A))

* 1. Matrix Rank, trace, det

A = np.array([[6, 1, 1],

              [4, -2, 5],

              [2, 8, 7]])

# Rank of a matrix

print("Rank of A:", np.linalg.matrix\_rank(A))

# Trace of matrix A

print("\nTrace of A:", np.trace(A))

# Determinant of a matrix

print("\nDeterminant of A:", np.linalg.det(A))

print("\nMatrix A raised to power 3:\n",

           np.linalg.matrix\_power(A, 3))

* 1. Broadcasting

When we do element-wise array operations, the shapes of the arrays have to match exactly. However, python has the ability of broadcasting, which can extend the dimension size from 1 to a number that matches the operator dimension.

1. Examples
2. array1 3x2x3

array2 1x2x3

When doing array1\*array2, array2 dimension automatically becomes 3x2x3.

1. Pandas
2. Read from csv

dataframe = pandas.read\_csv(‘csv\_file\_name’, engine = “python”)

1. Select rows and columns

1)Selecting columns using column key

dataframe[‘columnName’]

2)dataframe.columnName