karn Arora J078

ML ASSIGNMENT 2

Task 1 - Prove properties of matrix multiplication

```
import numpy as np

x = np.array([[1,2,3], [4,5,6], [7,8,9]])
y = np.array([[9,9,7], [7,6,5], [4,3,2]])
z = np.array([[1,1,1], [2,3,4], [5,7,9]])

I = np.identity(3)


print('Matrix A : \n', x)
print('Matrix B : \n', y)
print('Matrix C : \n', z)

print('Identity Matrix : \n', I)
```

```
Matrix A :
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
Matrix B :
 [[9 9 7]
 [7 6 5]
 [4 3 2]]
Matrix C :
 [[1 1 1]
 [2 3 4]
 [5 7 9]]
Identity Matrix :
 [[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

Communatitive property - not applicable

```
XdotY = x.dot(y)
YdotX = y.dot(x)

print('X.Y : \n', XdotY)
print('Y.X : \n', YdotX)
```

```
X.Y :
 [[ 35  30  23]
 [ 95  84  65]
```

```
      [155 138 107]]
     Y.X :
      [[ 94 119 144]
      [ 66  84 102]
      [ 30  39  48]]
```

Associative property [(A.B).C = A.(B.C)]

```
XY_Z = np.dot(x, y).dot(z)
X_YZ = x.dot(np.dot(y, z))

print('(A.B).C : \n', XY_Z)
print('A.(B.C) : \n', X_YZ)
```

```
     (A.B).C :
      [[ 210  286  362]
      [ 588  802 1016]
      [ 966 1318 1670]]
     A.(B.C) :
      [[ 210  286  362]
      [ 588  802 1016]
      [ 966 1318 1670]]
```

Distributive property [A.(B+C) =]

```
lhs = np.dot(x,y + z)
rhs = np.dot(x, y) + np.dot(x, z)
print("X.(Y+Z) : \n", lhs)
print('X.Y + X.Z : \n', rhs)
```

```
     X.(Y+Z) :
      [[ 55  58  59]
      [139 145 143]
      [223 232 227]]
     X.Y + X.Z :
      [[ 55  58  59]
      [139 145 143]
      [223 232 227]]
```

Identity property [A.I = I.A]

```
XI = np.dot(x, I)
IX = np.dot(I, x)

print('X.I : \n', XI)
print('I.X :\n', IX)
```

```
     X.I :
```

```
 [[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
I.X :
 [[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

Multiplicative property of zero [A.0 = 0.A = 0]

```
z_mat = np.zeros(9).reshape(3, 3)
lhs = np.dot(x, z_mat)
rhs = np.dot(z_mat, x)

print('X.0 : \n', lhs)
print('0.X : \n', rhs)
```

```
X.0 :
 [[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
0.X :
 [[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
```

Dimensions on matrix multiplication

```
p,q,r = 5,7,3
mat_p_q = np.random.randn(p, q)
mat_q_r = np.random.randn(q, r)
mat_mult = np.dot(mat_p_q, mat_q_r)
result_a, result_b = mat_mult.shape

print(f' {p}x{q} matrix X {q}x{r} matrix = {result_a}x{result_b} matrix')
```

```
 5x7 matrix X 7x3 matrix = 5x3 matrix
```

Task 2 - Inverse of a matrix

```
X_inv=np.linalg.inv(x)
X_inv
```

```
array([[ 3.15251974e+15, -6.30503948e+15,  3.15251974e+15],
       [-6.30503948e+15,  1.26100790e+16, -6.30503948e+15],
       [ 3.15251974e+15, -6.30503948e+15,  3.15251974e+15]])
```

Task 3 - Comparison of time between numpy and loops

```
import time

size = 5000

numpy_mat_X = np.random.randn(size, size)
numpy_mat_Y = np.random.randn(size, size)
list_mat_X = [list(i) for i in numpy_mat_X]
list_mat_Y = [list(i)for i in numpy_mat_Y]


start_loop = time.time()
list_mat_C = []
for i in range(size) :
    row = []
    for j in range(size) :
        row.append(list_mat_X[i][j] + list_mat_Y[i][j])
    list_mat_C.append(row)
end_loop = time.time()


start_numpy = time.time()
numpy_mat_C = numpy_mat_X + numpy_mat_Y
end_numpy = time.time()


print('Time for loops : ', end_loop - start_loop)
print('Time for numpy : ', end_numpy - start_numpy)
```

```
Time for loops :  20.611929655075073
Time for numpy :  1.2367503643035889
```

```
# Conclusioon - Numpy is faster than loops
```