Linear Algebra for Machine Learning

Reference: Chapter 2 (Linear Algebra) of the *Deep Learning Book* by Aaron Courville, Ian Goodfellow, and Yoshua Bengio (Attached)

Find the necessary files here > [CSE 472 Assignment 1 Files]

Task 1: Matrix Transformation

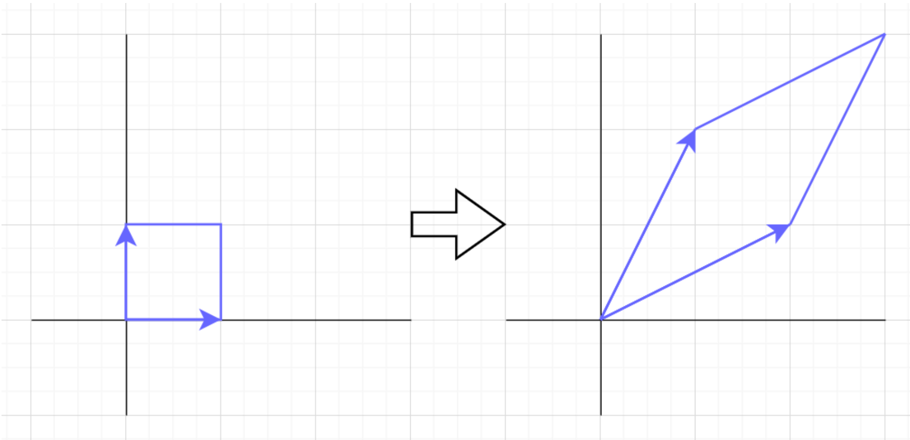
Go through and run the notebook “matrix-transformations-and-eigen-decomposition” to get an intuition about

- How a matrix can transform a vector?
- What do columns of matrices mean in terms of transformation?
- What does eigenvector mean?

(We recommend you also read the whole of Chapter 2 of the Deep Learning Book.)

Then,

- Change the cell values of matrix ***M*** so that it does the following shear transformation



- Run the whole notebook again and submit

Task 2: Eigen Decomposition

SubTask 2A: Random Matrix (random_eigen.py)

- Take the dimensions of matrix ***n*** as input.
- Produce a random ***n x n*** invertible matrix ***A***. For the purpose of demonstrating, every cell of ***A*** will be an integer.
- Perform Eigen Decomposition using NumPy’s library function
- Reconstruct ***A*** from eigenvalue and eigenvectors (Refer to Section 2.7).
- Check if the reconstruction is perfect. (`np.allclose` will come in handy)

SubTask 2B: Symmetric Matrix (symmetric_eigen.py)

- Take the dimensions of matrix ***n*** as input.
- Produce a random ***n x n*** invertible symmetric matrix ***A***. For the purpose of demonstrating, every cell of ***A*** will be an integer.
- Perform Eigen Decomposition using NumPy's library function
- Reconstruct ***A*** from eigenvalue and eigenvectors (Refer to Section 2.7).
- Check if the reconstruction is perfect. (`np.allclose` will come in handy)
- Please be mindful of applying efficient methods

Task 3: Singular Value Decomposition

(moore-penrose.py)

- Take the dimensions of matrix ***n, m*** as input.
- Produce a random ***n x m*** matrix ***A***. For the purpose of demonstrating, every cell of ***A*** must be an integer.
- Perform Singular Value Decomposition using NumPy's library function
- Calculate the Moore-Penrose Pseudoinverse using NumPy's library function
- Calculate the Moore-Penrose Pseudoinverse *again* using Eq. 2.47
- Check if these two inverses are equal (`np.allclose` will come in handy)

Submission

```
1705123
|-- matrix-transformations-and-eigen-decomposition.ipynb
|-- random_eigen.py
|-- symmetric_eigen.py
|-- moore-penrose.py
```

Zip the folder and rename it to **[Student_ID].zip**

Deadline: 02 December 2022, Friday 11.55 PM