Linear Algebra for Machine Learning

Reference:
1. Chapter 2 (Linear Algebra) of the "*Deep Learning Book*" by Aaron Courville, Ian Goodfellow, and Yoshua Bengio. deeplearningbook.org/contents/linear_algebra.html
2. "*Introduction to Linear Algebra for Applied Machine Learning with Python.*" 1 Aug. 2020, pabloinsente.github.io/intro-linear-algebra.

Courtesy: Md. Tareq Mahmood, Assistant Professor (on leave), CSE, BUET

Find the necessary files here > CSE 472 Assignment 1 Files

Task 1: Matrix Transformation

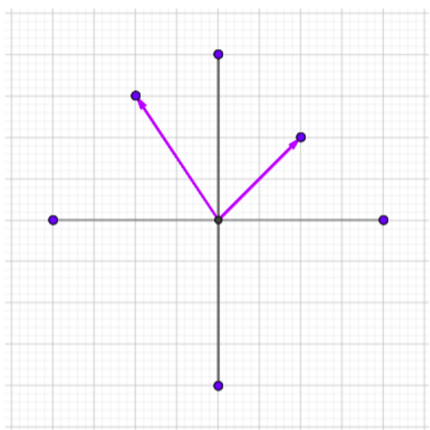
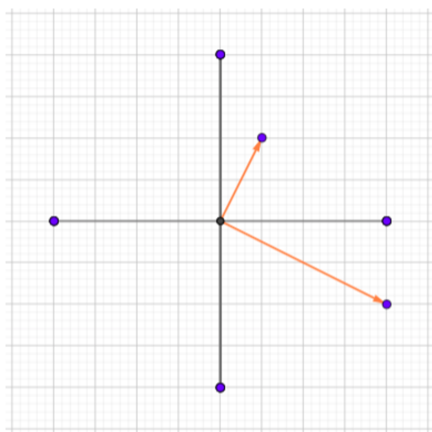
Go through and run the notebook "matrix-transformations-and-eigen-decomposition" to get an intuition about

- How can a matrix 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*** (there can be two such matrices; find one of them) so that it does the following transformation

Before Transformation	After 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 eigenvalues and eigenvectors (refer to Section 2.7).
- Check if the reconstruction worked properly. (np.allclose will come in handy.)
- You should be able to explain how your code ensures that the way you generated ***A*** ensures invertibility.

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 eigenvalues and eigenvectors (refer to Section 2.7).
- Check if the reconstruction worked properly. (np.allclose will come in handy.)
- Please be mindful of applying efficient methods (this will bear marks).
- You should be able to explain how your code ensures that the way you generated ***A*** ensures invertibility and symmetry.

Task 3: Image Reconstruction using Singular Value Decomposition

(image_reconstruction.py)

- Take a photo of a book's cover within your vicinity. Let's assume it is named `image.jpg`.
- Use OpenCV or similar frameworks to read `image.jpg.` Transform it to grayscale using functions such as `cv2.cvtColor()`. If you wish, resize to lower dimensions (~500) for faster computation.
- The grayscale image will be an ***n x m*** matrix ***A***.
- Perform Singular Value Decomposition using NumPy's library function.
- Given a matrix ***A*** and an integer ***k***, write a function `low_rank_approximation(A, k)` that returns the ***k***-rank approximation of ***A***.
- Now vary the value of ***k*** from 1 to ***min(n, m)*** (take at least 10 such values in the interval). In each case, plot the resultant ***k***-rank approximation as a grayscale image. Observe how the images vary with ***k***. You can find a sample intended output in the shared folder.

- Find the lowest ***k*** such that you can clearly read out the author's name from the image corresponding to the ***k***-rank approximation.

Marking Rubric

Task 1	20%
Task 2A	15%
Task 2B	15%
Task 3	50%

Since most of the tasks you have to do here are basically invocations of library functions, it is expected that you understand the underlying concepts properly to get full marks.

Submission

```
1805xyz
|-- matrix-transformations-and-eigen-decomposition.ipynb
|-- random_eigen.py
|-- symmetric_eigen.py
|-- image_reconstruction.py
|-- image.jpg
```

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

Deadline: November 25, 2023, Saturday, 10 PM