Imperial College Business School **(Due 06/13/2021)**

Machine Learning and Finance - Coursework -

Instructors: Arnaud de Servigny & Hachem Madmoun

Instructions:

- The code should be presented as a python file or a jupyter notebook.
- The theoretical questions can be answered in the jupyter notebook or in a separate pdf.

Notations:

- $\mathcal{M}_{n,p}(\mathbb{R})$ is the space of the matrices composed of n rows and p columns.
- $I_n \in \mathcal{M}_{n,n}(\mathbb{R})$ is the identity matrix of size n.
- For all $z \in \mathbb{R}^D$, the $\mathcal{L}^2$ norm on $\mathbb{R}^D$ of z is defined as follows: $||z||_2^2 = z^T z$
- For all $A = [a_{ij}]_{i,j} \in \mathcal{M}_{n,p}(\mathbb{R})$ we define the Frobenius norm of A as follows: $||A||_{\text{F}}^2 = \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2$
- The gradient of a function $f : \theta \in \mathbb{R}^D \mapsto \mathbb{R}$ at $\theta \in \mathbb{R}^D$ is denoted as follows $\nabla_\theta f(\theta) = \left(\frac{\partial f}{\partial \theta_1}(\theta), \ldots, \frac{\partial f}{\partial \theta_D}(\theta)\right)$
- Convention:
  - The rows $(A_i)_{1 \leq i \leq n}$ of a matrix $A = \begin{pmatrix} - & A_1 & - \\ \vdots & \vdots & \vdots \\ - & A_n & - \end{pmatrix} \in \mathcal{M}_{n,p}(\mathbb{R})$ are considered $\mathcal{M}_{p,1}(\mathbb{R})$ matrices.
  - The columns $(B_j)_{1 \leq j \leq p}$ of a matrix $B = \begin{pmatrix} | & \ldots & | \\ B_1 & \ldots & B_p \\ | & \ldots & | \end{pmatrix} \in \mathcal{M}_{n,p}(\mathbb{R})$ are considered $\mathcal{M}_{n,1}(\mathbb{R})$ matrices.

Presentation of the Coursework:

In this coursework, we are going to implement the GloVe approach. It was introduced by Jeffrey Pennington, Richard Socher and Christopher D. Manning in the paper: GloVe: Global Vectors for Word Representation.

The coursework is subdivided into three parts:

- In section 1, the objective is to load the data, preprocess it and create the **co-occurence matrix** (30 marks).
- In section 2, the objective is to train the model by using two different methods: **Gradient Descent** and **Alternating Least Squares** (50 marks).
- In section 3, the objective is to add a penalty term to the loss function as a **regularization** technique (20 marks).

1 Getting the statistics of the word occurences (30 marks)

1.1 Introducing the problem

The objective of the coursework is to train a model on a corpus of training sentences in order to represent words in a D-dimensional space. We would like to encode the similarity between the words in the embedding vectors themselves.

Question 1: Explain why this notion of similarity is not encoded in the one hot vector representation of words. (2 marks)

Several methods have been used to create word embeddings. The most popular ones rely on the intuition that *a word's meaning is given by the words that frequently appear close-by.*

For instance, we have introduced in Programming Session 6 the word2vec approach, which represents the tokens as parameters of a shallow neural network predicting a word's context given the world itself.

Although the shallow window-based model captures linguistic patterns between word vectors and performs well on the word analogy task ($w_{\text{France}} - w_{\text{Paris}} \approx w_{\text{England}} - w_{\text{London}}$), the model suffers from the disadvantage that they do not operate directly and the co-occurence statistics.

As explained in Lecture 6, the GloVe method is a popular method used to learn low-dimensional word representations by using **matrix factorization** methods on a matrix of word-word **co-occurence** statistics.

1.2 Preprocessing the data

The **data** folder contains a csv file named **RedditNews.csv**[1].

In the *RedditNews.csv* file are stored historical news headlines from Reddit WorldNews Channel, ranked by reddit users' votes, and only the top 25 headlines are considered for a single date.

You will find two colomns:

- The first column is for the "date".
- The second column is for the "News". As all the news are ranked from top to bottom, there are only 25 lines for each date.

Question 2: Load the data from the csv file, create a list of all the news. (2 marks)

Question 3: Preprocess the data by transforming the list of sentences into a list of sequences of integers, via a dictionary that maps the words to integers. (8 marks)

Question 4: For each sentence, add a specific index for the token "< sos >" (start of sequence) at the beginning of each sequence and an index for the token "< eos >" (end of sequence) at the end of each sequence.) (2 marks)

The resulting list of lists of integers is called **sequences**.

[1]Source: Sun, J. (2016, August) Daily News for Stock Market Prediction, Version 1. Retrieved [26 may 2020] from https://www.kaggle.com/aaron7sun/stocknews.

1.3 Creating the co-occurence matrix

Let V be the vocabulary size of the training corpus.

In Lecture 6, we have defined the co-occurence matrix $X = [X_{ij}]_{i,j} \in \mathcal{M}_{V,V}(\mathbb{R})$, whose entries X_{ij} represent the number of times word j appears in the context of word i.

Algorithm 1 summarizes the steps involved in estimating the co-occurence matrix from the corpus **sequences**.

Algorithm 1 Getting the co-occurence matrix

Input: sequences (list of lists of integers), context_size
Output: X (the co-occurence matrix)

Initialize the matrix $X \in \mathcal{M}_{V,V}(\mathbb{R})$ with zeros.
for sequence in sequences **do**
  for word $w[i]$ of index i in sequence **do**
    for word $w[j]$ of index j in the context of $w[i]$ **do**
      $X[w[i], w[j]] \longleftarrow X[w[i], w[j]] + 1$
    end for
  end for
end for

In algorithm 1, each time a word $w[j]$ (of index j in sequence) appears in the context of a center word $w[i]$ (of index i in sequence), we increase the value of $X[w[i], w[j]]$ by a value of 1 regardless of how close the word $w[j]$ is to the word $w[i]$.

We would like to take into consideration the distance $d(i, j)$ between the center word $w[i]$ and the context word $w[j]$ when updating the value $X[w[i], w[j]]$, as shown in figure 1

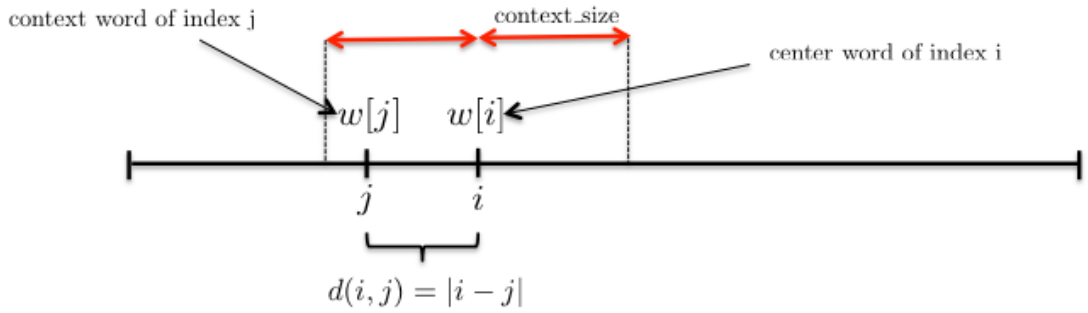

Figure 1

Question 5: Explain why it makes more sense to use the following update equation for $X[w[i], w[j]]$ when word $w[j]$ of index j is in the context word $w[i]$ of index i. (2 marks)

$$X[w[i], w[j]] \longleftarrow X[w[i], w[j]] + \frac{1}{|i - j|}$$

Question 6: Implement the new algorithm 2 to get the co-occurence matrix X. (12 marks)

Algorithm 2 Getting the co-occurence matrix

Input: sequences (list of lists of integers), context_size
Output: X (the co-occurence matrix)

```
Initialize the matrix X ∈ M_{V,V}(ℝ) with zeros.
for sequence in sequences do
    for word w[i] of index i in sequence do
        for word w[j] of index j in the context of w[i] do
            X[w[i], w[j]] ⟵ X[w[i], w[j]] + 1/|i−j|
        end for
    end for
end for
```

Since non-zero values in the matrix X are very large, we apply the logarithm function to all the elements of X (after adding 1 to all the entries X_{ij} to avoid applying the logarithm on zero values). The resulting matrix is still a sparse matrix. We will denote it $\log X$.

Question 7: Create the matrix $\log X$. (2 marks)

2 Training the weighted least squares regression model (50 marks)

2.1 Introducing the cost function

The logarithm of the co-occurence matrix $\log X$ has been defined in the previous section. The objective of this section is to approximate $\log X$ using a factorization method as follows:

$$\forall (i,j) \in \{1,\dots,V\}^2 \quad \log X_{ij} \approx W_i^T \tilde{W}_j + b_i + \tilde{b}_j$$

The parameters of the regression model are:

- A first **embedding matrix** and a bias term associated with it:

$$W = \begin{pmatrix} - & W_1 & - \\ \vdots & \vdots & \vdots \\ - & W_V & - \end{pmatrix} \in \mathcal{M}_{V,D}(\mathbb{R}), \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_V \end{pmatrix} \in \mathbb{R}^V$$

- A second **embedding matrix** and a bias term associated with it:

$$\tilde{W} = \begin{pmatrix} - & \tilde{W}_1 & - \\ \vdots & \vdots & \vdots \\ - & \tilde{W}_V & - \end{pmatrix} \in \mathcal{M}_{V,D}(\mathbb{R}), \quad \tilde{b} = \begin{pmatrix} \tilde{b}_1 \\ \vdots \\ \tilde{b}_V \end{pmatrix} \in \mathbb{R}^V$$

Instead of equal-weighting all the co-occurences, we introduce a **weighting function** $f(X_{ij})$ defined as follows:

$$\forall x \in \mathbb{R}_+ \quad f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } \ x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

The function f is represented in figure 2 with $x_{\max} = 100$ and $\alpha = 0.75$

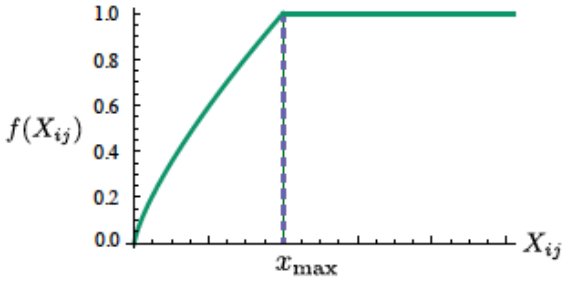

Figure 2: Weighting Function

Question 8: Create a matrix of shape (V, V) whose entries are $f(X_{ij})$. (6 marks)

Question 9: What are the hyperparameters associated with the weighting function and what is the intuition behind introducing it? (2 marks)

The **cost function** can then be written as follows:

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(\log X_{ij} - W_i^T \tilde{W}_j - b_i - \tilde{b}_j)^2$$

The gradients of the cost function J with respect to all the parameters are introduced in the following equations:

For all $i \in \{1, \ldots, V\}$ and all $j \in \{1, \ldots, V\}$:

$$\nabla_{W_i} J(W_i) = -2 \sum_{j'=1}^{V} f(X_{ij'}) \left(\log X_{ij'} - W_i^T \tilde{W}_{j'} - b_i - \tilde{b}_{j'} \right) \tilde{W}_{j'} \tag{2.1}$$

$$\nabla_{\tilde{W}_j} J(W_j) = -2 \sum_{i'=1}^{V} f(X_{i'j}) \left(\log X_{i'j} - W_{i'}^T \tilde{W}_j - b_{i'} - \tilde{b}_j \right) W_{i'} \tag{2.2}$$

$$\nabla_{b_i} J(b_i) = -2 \sum_{j'=1}^{V} f(X_{ij'}) \left(\log X_{ij'} - W_i^T \tilde{W}_{j'} - b_i - \tilde{b}_{j'} \right) \tag{2.3}$$

$$\nabla_{\tilde{b}_j} J(\tilde{b}_j) = -2 \sum_{i'=1}^{V} f(X_{i'j}) \left(\log X_{i'j} - W_{i'}^T \tilde{W}_j - b_{i'} - \tilde{b}_j \right) \tag{2.4}$$

Question 10: What is the total number of parameters in the model ? What are the shapes of all the gradients introduced in the equations 2.1, 2.2, 2.3 and 2.4 ? (2 marks)

Let us introduce two training methods:

- The first training method is called **alternating least squares**. It consists in finding the update equations by setting all the gradients to zero.
- The second training method consists in applying the **gradient descent** algorithm.

2.2 Alternating least squares

We would like to estimate the parameters $W, \tilde{W}, b, \tilde{b}$ by setting the gradients to zero.

Question 11: Show that: (2 marks)

$$\forall a, b \in \mathcal{M}_{D,1}(\mathbb{R}) \quad (a^T b)\ b = (b\ b^T)\ a$$

Question 12: By setting the gradients to zero, prove that we get the following update equations: (6 marks)

$$\nabla_{W_i} J(W_i) = 0 \iff W_i = \left(\sum_{j'=1}^{V} f(X_{ij'}) \tilde{W}_{j'} \tilde{W}_{j'}^T \right)^{-1} \left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - b_i - \tilde{b}_{j'}) \tilde{W}_{j'} \right)$$

$$\nabla_{\tilde{W}_j} J(\tilde{W}_j) = 0 \iff \tilde{W}_j = \left(\sum_{i'=1}^{V} f(X_{i'j}) W_{i'} W_{i'}^T \right)^{-1} \left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - b_{i'} - \tilde{b}_j) W_{i'} \right)$$

$$\nabla_{b_i} J(b_i) = 0 \iff b_i = \left(\sum_{j'=1}^{V} f(X_{ij'}) \right)^{-1} \left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - W_i^T \tilde{W}_{j'} - \tilde{b}_{j'}) \right)$$

$$\nabla_{\tilde{b}_j} J(\tilde{b}_j) = 0 \iff \tilde{b}_j = \left(\sum_{i'=1}^{V} f(X_{i'j}) \right)^{-1} \left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - W_{i'}^T \tilde{W}_j - b_{i'}) \right)$$

Each update equation for one parameter is a function of the other parameters. Therefore, in order to train our model, we can choose a number of iterations N_{epochs}, and apply the update equations N_{epochs} times by keeping track of the loss to make sure it converges.

For each iteration step $t \in \{0, \dots, N_{\text{epochs}} - 1\}$, let $W^{(t)}, \tilde{W}^{(t)}, b^{(t)}, \tilde{b}^{(t)}$ represent the parameters of our model at the iteration t.

The update equations from iteration t to $t+1$ can then be written as follows:

$$W_i^{(t+1)} \longleftarrow \left(\sum_{j'=1}^{V} f(X_{ij'}) \tilde{W}_{j'}^{(t)} \tilde{W}_{j'}^{(t)^T} \right)^{-1} \left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - b_i^{(t)} - \tilde{b}_{j'}^{(t)}) \tilde{W}_{j'}^{(t)} \right) \tag{2.5}$$

$$\tilde{W}_j^{(t+1)} \longleftarrow \left(\sum_{i'=1}^{V} f(X_{i'j}) W_{i'}^{(t)} W_{i'}^{(t)^T} \right)^{-1} \left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - b_{i'}^{(t)} - \tilde{b}_j^{(t)}) W_{i'}^{(t)} \right) \tag{2.6}$$

$$b_i^{(t+1)} \longleftarrow \left(\sum_{j'=1}^{V} f(X_{ij'}) \right)^{-1} \left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - W_i^{(t)^T} \tilde{W}_{j'}^{(t)} - \tilde{b}_{j'}^{(t)}) \right) \tag{2.7}$$

$$\tilde{b}_j^{(t+1)} \longleftarrow \left(\sum_{i'=1}^{V} f(X_{i'j}) \right)^{-1} \left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - W_{i'}^{(t)^T} \tilde{W}_j^{(t)} - b_{i'}^{(t)}) \right) \tag{2.8}$$

The pseudo code for the training algorithm can be expressed as follows:

Algorithm 3 Training by alternating least squares

Input: $\log X, f(X), N_{\text{epochs}}$
Output: $W^{(N_{\text{epochs}}-1)}, \tilde{W}^{(N_{\text{epochs}}-1)}, b^{(N_{\text{epochs}}-1)}, \tilde{b}^{(N_{\text{epochs}}-1)}$ (The trained paramaters)

Initialize randomly the parameters $W^{(0)}, \tilde{W}^{(0)}, b^{(0)}, \tilde{b}^{(0)}$
costs = [ ]
for $t \leftarrow 0, \ldots, N_{\text{epochs}} - 2$ **do**
    Calculate the cost as a function of $W^{(t)}, \tilde{W}^{(t)}, b^{(t)}, \tilde{b}^{(t)}$ and append the list costs
    for $i \leftarrow 0, \ldots, V-1$ **do**
        Update $W_i^{(t+1)}$ as a function of $\tilde{W}^{(t)}, b^{(t)}, \tilde{b}^{(t)}$ using 2.5
    end for
    for $j \leftarrow 0, \ldots, V-1$ **do**
        Update $\tilde{W}_j^{(t+1)}$ as a function of $W^{(t)}, b^{(t)}, \tilde{b}^{(t)}$ using 2.6
    end for
    for $i \leftarrow 0, \ldots, V-1$ **do**
        Update $b_i^{(t+1)}$ as a function of $W^{(t)}, \tilde{W}^{(t)}, \tilde{b}^{(t)}$ using 2.7
    end for
    for $j \leftarrow 0, \ldots, V-1$ **do**
        Update $\tilde{b}_j^{(t+1)}$ as a function of $W^{(t)}, \tilde{W}^{(t)}, b^{(t)}$ using 2.8
    end for
end for

Question 13: Implement the alternating least squares training algorithm 3. (14 marks)

Question 14: Plot the list of losses at the end of each iteration in algorithm 3. (2 marks)

2.3 Learning the weights using gradient descent

In this section, we would like to estimate the parameters of the model using gradient descent.

Let N_{epochs} be the number of epochs and η be the learning rate. We get the following training algorithm:

Algorithm 4 Training using gradient descent

Input: $\log X, f(X), \eta, N_{\text{epochs}}$
Output: $W^{(N_{\text{epochs}}-1)}, \tilde{W}^{(N_{\text{epochs}}-1)}, b^{(N_{\text{epochs}}-1)}, \tilde{b}^{(N_{\text{epochs}}-1)}$ (The trained paramaters)

Initialize randomly the parameters $W^{(0)}, \tilde{W}^{(0)}, b^{(0)}, \tilde{b}^{(0)}$
costs = [ ]
for $t \leftarrow 0, \ldots, N_{\text{epochs}} - 2$ **do**
    Calculate the cost as a function of $W^{(t)}, \tilde{W}^{(t)}, b^{(t)}, \tilde{b}^{(t)}$ and append the list costs
    for $i \leftarrow 0, \ldots, V-1$ **do**
        $W_i^{(t+1)} \longleftarrow W_i^{(t)} - \eta \nabla_{W_i} J(W_i^{(t)})$
    end for
    for $j \leftarrow 0, \ldots, V-1$ **do**
        $\tilde{W}_j^{(t+1)} \longleftarrow \tilde{W}_j^{(t)} - \eta \nabla_{\tilde{W}_j} J(\tilde{W}_j^{(t+1)})$
    end for
    for $i \leftarrow 0, \ldots, V-1$ **do**
        $b_i^{(t+1)} \longleftarrow b_i^{(t)} - \eta \nabla_{b_i} J(b_i^{(t)})$
    end for
    for $j \leftarrow 0, \ldots, V-1$ **do**
        $\tilde{b}_j^{(t+1)} \longleftarrow \tilde{b}_j^{(t)} - \eta \nabla_{\tilde{b}_j} J(\tilde{b}_j^{(t)})$
    end for
end for

Question 15: Implement the gradient descent training algorithm 4. (14 marks)

Question 16: Plot the list of losses at the end of each iteration in algorithm 4. (2 marks)

3 Introducing regularization (20 marks)

Let us introduce a regularization penalty term in the cost function. The new cost function is defined as follows:

$$\tilde{J} = \sum_{i=1}^{V}\sum_{j=1}^{V} f(X_{ij})(\log X_{ij} - W_i^T\tilde{W}_j - b_i - \tilde{b}_j)^2 + \lambda\left(||W||_{\mathrm{F}}^2 + ||\tilde{W}||_{\mathrm{F}}^2 + ||b||_2^2 + ||\tilde{b}||_2^2\right)$$

Question 17: Show that: $||W||_{\mathbf{F}}^2 = \sum_{i=1}^{V} W_i^T W_i$ (2 marks)

Question 18: Deduce that for all $i \in \{1, \ldots, V\}$**:** (4 marks)

$$\nabla_{W_i}(||W||_{\mathrm{F}}^2) = 2W_i \tag{3.1}$$
$$(\text{Hint} : \forall z \in \mathbb{R}^D \ \forall A \in \mathcal{M}_{D,D}(\mathbb{R}) \quad \nabla_z(z^T A z) = (A + A^T)z$$

Question 19: From the equations 2.1, 2.2, 2.3, 2.4 and 3.1, show that the update equations for the method of alternating least squares become: (8 marks)

$$W_i^{(t+1)} \longleftarrow \left(\sum_{j'=1}^{V} f(X_{ij'})\tilde{W}_{j'}^{(t)}\tilde{W}_{j'}^{(t)^T} + \lambda I_D\right)^{-1}\left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - b_i^{(t)} - \tilde{b}_{j'}^{(t)})\tilde{W}_{j'}^{(t)}\right)$$
$$\tilde{W}_j^{(t+1)} \longleftarrow \left(\sum_{i'=1}^{V} f(X_{i'j})W_{i'}^{(t)}W_{i'}^{(t)^T} + \lambda I_D\right)^{-1}\left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - b_{i'}^{(t)} - \tilde{b}_j^{(t)})W_{i'}^{(t)}\right)$$
$$b_i^{(t+1)} \longleftarrow \left(\sum_{j'=1}^{V} f(X_{ij'}) + \lambda\right)^{-1}\left(\sum_{j'=1}^{V} f(X_{ij'})(\log X_{ij'} - W_i^{(t)^T}\tilde{W}_{j'}^{(t)} - \tilde{b}_{j'}^{(t)})\right)$$
$$\tilde{b}_j^{(t+1)} \longleftarrow \left(\sum_{i'=1}^{V} f(X_{i'j}) + \lambda\right)^{-1}\left(\sum_{i'=1}^{V} f(X_{i'j})(\log X_{i'j} - W_{i'}^{(t)^T}\tilde{W}_j^{(t)} - b_{i'}^{(t)})\right)$$

Question 20: What would be the update equations for minimizing the new loss function $\tilde{J}$ **by using the gradient descent algorithm.** (6 marks)