# Introduction to Machine Learning (67577)

## Exercise 1
## Mathematical Background

Second Semester, 2021

## Contents

# 1 Theoretical

## 1.1 Linear Algebra

### 1.1.1 Recap

1. Calculate the projection of $v = (1,2,3,4)^\top$ on the vector $w = (0,-1,1,2)^\top$.

$$\begin{aligned}\left\langle \begin{pmatrix}1\\2\\3\\4\end{pmatrix}, \begin{pmatrix}0\\-1\\1\\2\end{pmatrix} \right\rangle &= (1\cdot 0 + 2\cdot -1 + 3\cdot 1 + 4\cdot 2) \\ &= (-2+3+8) = 9 \\ \left|\left|(0,-1,1,2)^\top\right|\right|^2 &= 6\end{aligned}$$

So we get: $\frac{3}{2}\cdot(0,-1,1,2)^\top$

2. Calculate the projection of $v = (1,2,3,4)^\top$ on the vector $w = (1,0,1,-1)^\top$.

$$\left\langle \begin{pmatrix}1\\2\\3\\4\end{pmatrix}, \begin{pmatrix}1\\0\\1\\-1\end{pmatrix} \right\rangle = 1+0+3-4 = 0$$

$$\left|\left|(1,0,1,-1)^\top\right|\right| = 3$$

So we get: $0\cdot(1,0,1,-1)^\top = 0$

3. Prove the angle between two non-zero vectors $v, w \in \mathbb{R}^m$ is $\pm 90^\circ$ iff $v^\top w = 0$.

As seen in recitation: $\langle x,y\rangle = ||x||\cdot||y||\cdot\cos\theta$. Since $||x||, ||y|| > 0$, we get

$$\langle x,y\rangle = 0 \Leftrightarrow \cos\theta = 0 \Leftrightarrow x \perp y$$

4. Prove that orthogonal matrices are isometric transformations. That is let $T : V \mapsto W$ be some linear transformation and $A$ the corresponding matrix. Then if $A$ is orthogonal then $\forall x \in V \;\; ||Ax|| = ||x||$.

$$||Ax||^2 = (Ax)^\top (Ax) = x^\top A^\top A x = x^\top I x = ||x||^2$$

### 1.1.2 Singular Value Decomposition

5. Let $A$ be an invertible matrix. Write a formula for the inverse of $A$ using only the matrices of the SVD of $A$: $U\Sigma V^\top$. Explain why knowing the SVD of matrix is useful in this context.

As $\Sigma$ a diagonal matrix it is easy to compute its inverse:

$$\Sigma^{-1} = \begin{bmatrix} \Sigma_{11}^{-1} & & 0 \\ & \ddots & \\ 0 & & \Sigma_{dd}^{-1} \end{bmatrix}$$

Therefore:

$$A^{-1} = V\Sigma^{-1}U^\top = V \begin{bmatrix} \Sigma_{11}^{-1} & & 0 \\ & \ddots & \\ 0 & & \Sigma_{dd}^{-1} \end{bmatrix} U^\top$$

6. Compute the SVD of

$$A = U\Sigma V^\top = \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix}$$

That is, find the matrices $U, \Sigma, V^\top$ where $U, V$ are orthogonal matrices and $\Sigma$ is diagonal. Guidance:

- Calculate $A^\top A$
- Deduce $V$ and $\Sigma$. Use the *Eigenvalues Decomposition*, either manually (good practice) or use numpy.linalg.eig
- Find $U$ using the equality $AV = U\Sigma$

$$U = \begin{bmatrix} \sqrt{\frac{1}{2}} & \sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \end{bmatrix} \quad \Sigma = \begin{bmatrix} \sqrt{80} & 0 \\ 0 & \sqrt{20} \end{bmatrix} \quad V^\top = \begin{bmatrix} \sqrt{\frac{1}{10}} & \sqrt{\frac{9}{10}} \\ \sqrt{\frac{9}{10}} & -\sqrt{\frac{1}{10}} \end{bmatrix}$$

7. (Power Iteration) In this section we will implement an algorithm for SVD decomposition, we will use the relation between SVD of $A$ to EVD of $A^T A$ that we saw in recitation. For some $A \in M_{m\times n}(\mathbb{R})$, define $C_0 = A^T A$.
Let $\lambda_1, \lambda_2, ...\lambda_n$ be the eigenvalues of $C_0$, with the corresponding normalized eigenvectors $v_1, v_2, ...v_n$, ordered such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$.
Assume $\lambda_1 > \lambda_2$, where $\lambda_1$ is the largest eigenvalue and $\lambda_2$ is the second-largest one.
Define $b_{k+1} = \frac{C_0 b_k}{\|C_0 b_k\|}$, and let $b_0 = \sum_{i=1}^n a_i v_i$, where $a_1 \neq 0$.
Show that: $\lim_{k\to\infty} b_k = \pm v_1$.

Define $b_0 := \sum c_i v_i$. Then:

$$\begin{aligned} A^k b_0 &= \sum c_i \cdot A^k v_i \\ &= \sum c_i A^{k-1}(Av_i) \\ &= \sum c_i A^{k-1}(\lambda_i v_i) \\ &= \sum c_i \lambda_i^k v_i \\ &= c_1 \lambda_1^k \left( v_1 + \sum_{i=2} \frac{c_i}{c_1} \left(\frac{\lambda_i}{\lambda_1}\right)^k v_i \right) \end{aligned}$$

Therefore, as we increase $k$:

$$\begin{aligned}\lim_{k\to\infty} A^k b_0 &= \lim_{k\to\infty} c_1\lambda_1^k v_1 + c_1\lambda_1^k \textstyle\sum_{i=2} \lim_{k\to\infty}\left(\frac{\lambda_i}{\lambda_1}\right)^k c_i v_i \\ &= c_1\lambda^k v_1 + c_1\lambda_1^k \textstyle\sum_{i=2} 0\cdot c_i v_i \\ &= c_1\lambda^k v_1\end{aligned}$$

As for any $i > 1$ it holds that $(\lambda_i/\lambda_1) < 1$.

## 1.2 Multivariate Calculus

8. Let $x \in \mathbb{R}^n$ be a fixed vector and $U \in \mathbb{R}^{n\times n}$ a fixed orthogonal matrix. Calculate the Jacobian of the function $f : \mathbb{R}^n \to \mathbb{R}^n$:

$$f(\sigma) = U \cdot diag(\sigma) U^\top x$$

Where $diag(\sigma)$ is an $n \times n$ matrix where

$$diag(\sigma)_{ij} = \begin{cases}\sigma_i & i = j \\ 0 & i \neq j\end{cases}$$

We can express $f(\sigma)$ as

$$f(\sigma) = \sum_{i=1}^{n} \sigma_i u_i u_i^\top x,$$

where $u_i$ is the $i$'th column of $U$. From here it's easy to see that $\frac{\partial f_j(\sigma_0)}{\partial \sigma_i} = [u_i u_i^\top x]_j$, which means the $i$ 'th column of $J_\sigma(f)$ is $u_i \cdot \langle u_i, x\rangle$. In matrix notation we can write $J_\sigma(f) = U \cdot diag\left(U^\top x\right)$

9. Use the chain rule to calculate the gradient of $h(\sigma) = \frac{1}{2}||f(\sigma) - y||^2$

$$\nabla h = (f(\sigma) - y)^T \nabla f(\sigma) \cdot \overrightarrow{1}$$

10. Calculate the Jacobian of the softmax function (initial steps can be found in recitation file):

$$g(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

$$\frac{\partial g_i(z)}{\partial z_j} = g_i(z)(\delta_{ij} - g_j(z))$$

# 2 Practical

## 2.1 Multivariate Gaussians

In this question we will examine the three-dimensional case. We want to show how linear transformations affect the data set and in result the covariance matrix. First we will generate random points with mean values at the origin and unit variance $\sigma(x) = \sigma(y) = \sigma(z) = 1$.

11. Download the file 3d_gaussian.py. Use the identity matrix as the covariance matrix to generate random points and than plot them (with the given function). *You are not obligated to use or submit the "3d_gaussian.py" file. It is there to help you start*
12. Transform the data with the following scaling matrix:

$$S = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

    Plot the new points. What does the covariance matrix look like now (both analytically and numerically)?
13. Multiply the scaled data by random orthogonal matrix. Plot the new points. What does the covariance matrix look like now?
14. In recitation we claimed that the marginal distribution of a gaussian is still gaussian. Plot the projection of the data to the x, y axes. What does it look like? Add the plot to the submission.
15. In recitation we claimed that the conditional distribution of a gaussian is still gaussian. Only for points where $0.1 > z > -0.4$: Plot the projection of the points to the x, y axes. What does it look like? Add the plot to the submission.

## 2.2 Concentration Inequalities

In the following weeks we will introduce the PAC model for learnability, that tells us when a given hypothesis class is learnable. It describes a relationship between 3 factors: the sample complexity ($m$, number of samples in our dataset), our approximation of the learner's loss ($\varepsilon$) and our confidence level ($\delta$). **Intuitively speaking**, given a dataset of size $m$ drawn from some distribution $\mathcal{D}$, we want "with high probability" ($1-\delta$) that the learner's loss over that dataset will be very small ($\varepsilon$). In the following question we will show some aspects of the relationship between the factors. You will not need (yet) any deep understanding of this model, but just the probabilistic bounds seen in class.

16. Consider the following scenario. Let $X$ be a Bernoulli random variable representing a coin toss, with probability of $\varepsilon$ being 1. You are given $100,000$ sequences of $1,000$ coin tosses (arranged in a matrix, "data", of 100000 rows and 1000 columns. That is, each coin toss is like:

$$X \sim Ber(\varepsilon)$$

    and all coin tosses can be though of as:

$$\forall j \in [100000] \quad X_1^j, \ldots, X_{1000}^j \overset{iid}{\sim} Ber(\varepsilon)$$

    where *iid* means that all have the same distribution (Bernoulli $\varepsilon$) and are independent ($X_l^j, X_k^j$ are independent $\forall l \neq k$). In addition tosses of different sequences (and therefore the sequences too) are independent.

    Using this scenario we explore the Chebyshev and Hoeffding bounds on our estimation of $\varepsilon$. To generate the data use the commands:
    - import numpy
    - data = numpy.random.binomial(1, epsilon, (NSAMPLES,NTOSSES))

    where epsilon, NSAMPLES and NTOSSES get values as described below.

(a) Generate a dataset as described above where $X \sim Ber(0.25)$. For the first 5 sequences of 1000 tosses (the first 5 rows in "data"), plot the estimate $\overline{X}_m = \frac{1}{m}\sum_{i=1}^{m} X_i$ as a function of $m$ (i.e the mean of all tosses up to $m$).
You are expected to create 1 figure with 5 plots (each row of the dataset should be plotted in a different color). What do you expect to see in this plot as $m$ grows?

Define a variable epsilon which gets the values $[0.5, 0.25, 0.1, 0.01, 0.001]$. For each value of epsilon generate a dataset and perform the tasks below.

(b) For each bound (Chebyshev and Hoeffding seen in class) and for each $\varepsilon$, plot the upper bound on $\mathbb{P}_{X_1,\ldots,X_m}\left(\left|\overline{X}_m - \mathbb{E}[X]\right| \geq \varepsilon\right)$ (derived in class) as a function of $m$ (where $m$ ranges from 1 to 1000). $X$ refers to the random variable seen in the explanation above.

You are expected to create 5 figures with 2 plots each (mention in the title of each plot what is $\varepsilon$ and use a different color for each bound)[1].

(c) You are now told that $p = 0.25$. On top of the figures from the previous question, plot the percentage of sequences that satisfy $\left|\overline{X}_m - \mathbb{E}[X]\right| \geq \varepsilon$ as a function of $m$, where this time $\mathbb{E}[X] = p = 0.25$. What are you expecting to see in these plots? Explain. In your submission be sure to add plots of clause $(b)$ and of clause $(c)$.

[1] if the bound calculated is greater than 1, you should substitute it with 1. This is because we are bounding a probability which is always smaller than 1.