UiT

THE ARCTIC UNIVERSITY OF NORWAY

1. Vectorising Code and Machine Learning Basics

FYS-2021 Exercises

Department of Physics and Technology
Faculty of Science and Technology

Vectorising code

Problem 1

In previous years, our exercises have been centered on providing a practical experience with various machine learning methods. This semester, we are taking a step further to examine the underlying rationale behind these methods through open discussions. In an effort to foster transparent and collaborative learning, Canvas will serve as our discussion platform, facilitating constructive dialogue among students, lecturers, and exercise instructors.

We warmly encourage you to upload your exercise answers to Canvas. Although this is not a requirement, your active participation significantly helps us enhance the quality of our course. Furthermore, we recommend leaving positive and constructive feedback on your classmates' exercises. Your insights and collaboration are extremely valuable to us!

Additionally, this exercise aims to enhance your visualization skills, a critical tool in both understanding machine learning concepts and effectively communicating them to your readers and colleagues. Hence, for the relevant problems, your responses should consist of visual representations that address the questions, accompanied by brief captions to explain your illustrations. Focus on simplicity in your visualizations, aiming to convey the key message. Sketching by hand is entirely acceptable. Look to relevant materials for inspiration.

With regards to the hands-on problems, we highly recommend Python as the programming language for this course. While students are free to choose other programming languages, please note that our exercise instructors may not be able to provide detailed coding assistance for languages other than Python.

(1a) For students who choose to use Python, but do not have much experience with the numerical library NumPy (or want to refresh their knowledge), it is recommended to do a few of the "warm-up" exercises that you can find here:

https://www.w3resource.com/python-exercises/numpy/basic/index.php

https://www.w3resource.com/python-exercises/numpy/index-array.php

https://www.w3resource.com/python-exercises/numpy/linear-algebra/index.php

https://www.w3resource.com/python-exercises/numpy/python-numpy-stat.php

Students who are familiar with Numpy and students who choose to use other programming languages can move on to the exercises below.

This course assumes you have a fair knowledge of how numerical programming, (i.e. working with vectors and matrices programatically), works. In the following exercises, we will look at writing efficient, vectorised code. Assume that we have vectors

$$\boldsymbol{a} \in \mathbb{R}^n \text{ with elements } \boldsymbol{a} = [a_1, \ldots, a_n]^T$$

$$\boldsymbol{b} \in \mathbb{R}^n \text{ with elements } \boldsymbol{b} = [b_1, \ldots, b_n]^{\mathrm{T}}$$

Further assume we have a matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1N} \\ & & \vdots & \\ x_{M1} & x_{M2} & \ldots & x_{MN} \end{bmatrix}$$

The result vector is $\boldsymbol{y}$ which takes its dimensionality based on the problem.

In all problems you should assume that the dimensions of the vectors and matrices involved are such that the matrix multiplications (and inner products) are valid. Note that e.g. `y+=k` is the same as `y=y+k`.

(1b) Write this sum as a mathematical vector operation

$$y = \sum_{i=1}^{n} a_i b_i$$

(1c) Write this for loop as a mathematical vector operation and as vectorised code.

```
for i=1,...,n:
  y+=a_i b_i
```

(1d) Write this for loop as a mathematical vector operation and vectorised code (assume $N = n$)

```
for i=1,...,M:
  for k=1,...,n:
    y_i+=x_ik a_k
```

(1e) *Vectorise* the following algorithm (answer with code). This shouldn't be done with traditional matrix multiplications *alone*:

```
for i=1,...,M:
  for k=1,...,n:
    y+=x_ik a_k
```

(1f) Let $Z \in \mathbb{R}^{P \times M}$ and $z_{ij} \in Z$ be the entry in row i and column j.

Vectorise the following algorithm

```
for j=1,...,N:
  for i=1,...,P:
    for k=1,...,M:
      y_i+=z_ik x_kj
```

(1g) NB:*This part problem has different notation!*
Suppose you have a dataset with N samples and n features:

$$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N, \boldsymbol{x}_i = [x_{i1}, \ldots, x_{in}] \in \mathbb{R}^n$$

Similarly, you have parameter vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_P$, $\boldsymbol{w} \in \mathbb{R}^n$ and $\boldsymbol{a} \in \mathbb{R}^P$.

Implement an algorithm, *for every single sample* $\boldsymbol{x}_j = [x_{j1}, \ldots, x_{jn}]$ in the dataset, that calculates:

$$v_i = \sum_{k=1}^{n} w_{ik} x_{jk}$$

$$y_j = \sum_{i=1}^{P} v_i a_i$$

Vectorise these expressions. You want to end up with a vector $\boldsymbol{y} \in \mathbb{R}^N$

Because of the potential lack of relevance, the next problems are considered 'bonus' exercises.

(1h) Implement an algorithm that does matrix multiplication (without using built-in matrix multiplication functions).

(1i) Show that for any matrix $\mathbf{A}$ the product $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is symmetric.

(1j) Implement a algorithm that calculates $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ using the property of symmetry. Generate a huge matrix $\mathbf{A}$ and compare the running time between the implementation you did in (1h) and the built in matrix multiplication function.

(1k) Provide a brief opinion, encapsulated in one or two sentences, on why you believe vectorised (or matrix-based) computations hold an advantage over traditional 'for' loops.

Machine learning basics

Problem 2

(2a) Create a visual comparison between predictive and generative models. In the caption, clearly outline their differences, enumerate reasons for distinguishing between them, and list practical applications for each approach in the field of machine learning.

For your understanding, this course mainly concentrates on predictive models, whereas generative models are extensively covered in more advanced courses, such as Deep Learning (FYS-3033).

(2b) Assume that you have split your collected dataset into three subsets: a training set $\mathbf{X}_{tr}$, a validation set $\mathbf{X}_{val}$, and a test set $\mathbf{X}_{te}$. Your training set $\mathbf{X}_{tr}$ consists of five instances, i.e., $\mathbf{X}_{tr} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_5, y_5)\}$, where each $\mathbf{x}_n$ is the multivariate input and y_n is the corresponding ground-truth label. You intend to train your model $f(\cdot)$ in a supervised manner for a predictive task, such that a prediction $\hat{y}_n$ is the model output given by $\hat{y}_n = f(\mathbf{x}_n)$.

Visualise the process of supervised learning, incorporating the five training instances mentioned earlier. In addition, provide two distinct visuals comparing supervised learning with 1) unsupervised learning (also known as self-supervised learning) and 2) semi-supervised learning.

Problem 3

Classification is a common predictive task, where the aim is to set the decision boundaries separating different classes. As an illustrative example, we explore a multivariate binary classification task, specifically, distinguishing between dogs and cats.

(3a) Identify five features that can effectively distinguish between dogs and cats based on high and low values. For example, consider the feature 'preference for solitude'. A high value might be indicative of a cat, which often enjoys being alone, while a low value might suggest a dog, known for being more social.

(3b) Assume you standardise each feature to have within the range $[0, 1]$.

Assign a unique threshold value for each feature you have created in (3a). These thresholds will function as decision boundaries for your classifier. These thresholds should be set based on your personal understanding and insights, rather than defaulting to the straightforward choice of a 0.5 midpoint. Keep in mind that a score higher than the threshold represents cats, while a score below the threshold signifies a dog. Should the scoring range for cats and dogs require inversion on the y-axis, apply a negation to the corresponding feature. Consequently, you will formulate a vector $\mathbf{x} = [f_1, \ldots, f_5]^T$, each element of which represents the respective threshold for a feature.

(3c) Plot your threshold vector $\mathbf{x}$ on a Cartesian coordinate. The x-axis should represent your five features, $[f_1, \ldots, f_5]$. Each corresponding threshold value should be indicated on the y-axis, which should range from 0 at the bottom to 1 at the top. Next, connect the threshold values to form a decision boundary line on the graph. This line will partition the Cartesian plane into two distinct regions: the upper region, signifying cats, and the lower region, indicative of dogs.

(3d) Suppose you have chosen features that are plausible and carry sufficient discriminative power for classification, differentiating classes around the 0.5 threshold mark. The figure below illustrates a classifier that learns an individual's perceptions of these features, exhibiting a characteristic of overfitting.

Elaborate on why this classifier is overfitted, and explore the potential risks it could pose.